A METHODOLOGY FOR NON-INTRUSIVE PROJECTION-BASED MODEL REDUCTION OF EXPENSIVE BLACK-BOX PDE-BASED SYSTEMS AND APPLICATION IN THE MANY-QUERY CONTEXT

A Dissertation
Presented to
The Academic Faculty

By

Sudharshan Ashwin Renganathan

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Georgia Institute of Technology

Georgia Institute of Technology

May 2018

A METHODOLOGY FOR NON-INTRUSIVE PROJECTION-BASED MODEL REDUCTION OF EXPENSIVE BLACK-BOX PDE-BASED SYSTEMS AND APPLICATION IN THE MANY-QUERY CONTEXT

Approved by:

Dr. Dimitri N. Mavris, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Graeme J. Kennedy
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Daniel Schrage
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Steven Berguin
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Juan J. Alonso
School of Aerospace Engineering
Stanford University

Date Approved: April 5, 2018

I have had my results for a long time: but I do not yet know how I am to arrive at them.

- *Carl Friedrich Gauss*

Dedicated to my parents, Mrs and Mr Renganathan

ACKNOWLEDGEMENTS

I want to begin by thanking my parents, Mrs Kamala Renganathan and Mr Desikan Renganathan for bringing me into this world and showering me with their unconditional love and affection. My mother's strong-willed personality served as an inspiration and taught me how to pursue goals in the face of adversity. Without my father's hard work and sacrifice, a life with high-quality education and sufficient resources would not have been possible.

I sincerely thank my Ph.D. advisor and mentor, Professor Dimitri Mavris for accepting me into the Aerospace Systems Design Laboratory, financially supporting my entire stay as a Ph.D. student, offering me the right advice during every stage of the program, showing limitless patience and never giving up on me, providing the best atmosphere in the lab to draw inspiration from and grow, and always standing tall and setting the right example for me to follow. It has been a pleasure and true honor to be earn my Ph.D. by being part of the ASDL.

I want to extend thanks to my thesis reading committee; specifically to Professors Graeme Kennedy and Daniel Schrage for being part of core committee over the past 2 years. Special thanks to Professor Juan J. Alonso for graciously accepting to participate in my thesis defense despite his busy schedule. I believe that the cameo he played towards the end added lot of value to my thesis. I thank Dr Steven Berguin for his careful analysis of this thesis and offering insightful feedback. I have enjoyed and greatly value his association both through the Airbus OWN project and my thesis over the past year and look forward to more collaboration in the future. Although not part of the committee, I thank Professor Yingjie Liu for the many meetings and lengthy discussions we had, that helped me build the foundation for my thesis.

I acknowledge and value the association with several other professors at Georgia Tech. Thanks to Professor Suresh Menon for accepting me into his group and supporting me for

the masters degree. I thank Professor Jerry Seitzman who's teaching method I found to be exceptional and most impactful, and Professor Narayanan Komerath who triggered my interest in aerodynamics. Thanks to Professor Jechiel Jagoda for the warmth and kindness he showed whenever I approached him with questions on academic matters.

Amongst all my teachers and mentors, I owe the most thanks to my high school physics teacher Mr N Nagarajan. He had sown the seeds of scientific curiosity and the love for math, physics & chemistry in me and is solely responsible for launching me on a trajectory that would eventually lead me to earning a Ph.D. from GeorgiaTech. I cannot forget the lengthy evening chats I have had with him at the physics lab, discussing specific problems from *I.E.Irodov's* book and the *Physics Today* magazine, and walking back home together after fulfilling the lab assistant's role of turning-off the lights and shutting down doors and windows. His elegance in building up various concepts from first principles, optimally blending formal and informal methods of teaching and introducing mathematics as a *language of physics* had left a lasting impact in me.

I would be remiss not to acknowledge the contributions of friends and fellow graduate students both at the ASDL and elsewhere, although the list of names is too long to fit in here. I consider myself lucky to have been able to work with some gifted students, both junior and senior to me in the academic hierarchy, from whom I have developed a lot of problem solving skills that are not necessarily taught in the class room. I can only hope that my association has offered them something useful in return. I appreciate the help of the senior members for their help with preparing for the qualifying exams and the thesis proposal. A special mention goes to Mr Russell Denney who has helped me out on numerous occasions when I ran into issues with NPSS or anything turbomachinery related. It has been an honor to have worked with such a knowledgeable and yet down-to-earth person. Special thanks to Arun Ramamurthy and Dushhyanth Rajaram for their company during the much needed coffee breaks!

I want to thank my wife Supraja for her companionship that encompasses love, friend-

ship, camaraderie, constructive criticism, moral support and a deeper understanding of me that is unparalleled. She is the only person who was privy to all of the ups and downs through my stay at GeorgiaTech. Our association goes back to our undergraduate days and both of us have grown and transformed leaps and bounds since then; but the bonding has remained strong all through. I also thank my sisters Srimathi and Dharini for their warmth and support, and my nieces-and-nephews *Athithi, Ananya, Hayagriv* and *Saketh*, for offering me priceless joy during my visits.

Finally, thanks to the spiritual preachings of *Swami Vivekananda* and *Carnatic* music soulfully rendered by eminent musicians such as (and in no particular order) *Semmangudi Srinivasa Iyer, G.N.Balasubramaniam, M.S.Subbulakshmi, Ranjani & Gayathri* and *Sanjay Subrahmanyam*, for making my life at GeorgiaTech a lot smoother than it could have been.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

The projection-based reduced order modeling, typically requires access to the discrete form of governing equations of the high-fidelity model. The projection is commonly done on a subspace determined via POD. However when commercial codes are used as the high-fidelity model, such an approach is not possible in general. Usually in such circumstances, a POD+Interpolation approach is taken where the reduced state variable is directly interpolated to adapt for change in time/parameters. This thesis devices a method to develop projection-based ROM with commercial codes, specifically CFD codes. The novelty of the work is that it converts the original non-linear PDE system into a linear PDE system with auxiliary non-linear algebraic equations which are then projected onto the POD subspace. By such a linearization, it is shown that the governing equations can be extracted by directly discretizing the linear terms (which is easier compared to non-linear terms) at a computational cost that scales linearly with grid size (N). Other methods that exist to discover governing equations from data, are known to also involve a similar or higher cost, while being tailored towards time-dependent systems. Finally, the ROM is posed as a constrained optimization problem that can be solved cheaply. Since the thesis specifically addresses static parametric systems, a database of such ROMs are generated for a pre-determined set of parameter snapshots which are then interpolated by mapping them to the tangent space of the manifold they are embedded in (manifold of symmetric positive definite matrices in this case) to adapt for parametric changes. The method is tested on canonical PDEs and flow past airfoils at subsonic and transonic flow regimes. A prediction error of $< 5\%$ was achieved in subsonic cases in terms of the state, pressure distributions, lift and drag. Under transonic conditions with moving shocks, the approach incurs higher error unless a sufficiently dense snapshot distribution is used. Model parameters are identified and experiments are conducted to determine settings that improve accuracy. The usefulness of the method is also demonstrated on application problems in the many-query context - design

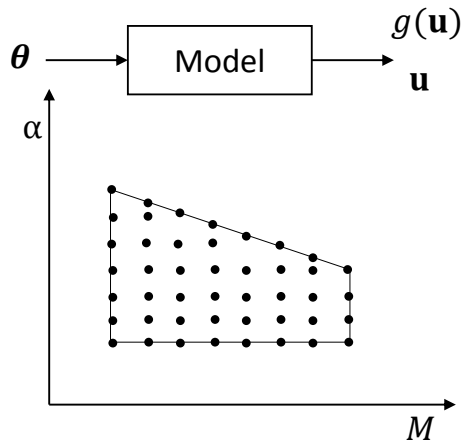
optimization and uncertainty quantification. Overall, the strength and weaknesses of the approach are identified, demonstrated and explained.

CHAPTER 1
INTRODUCTION

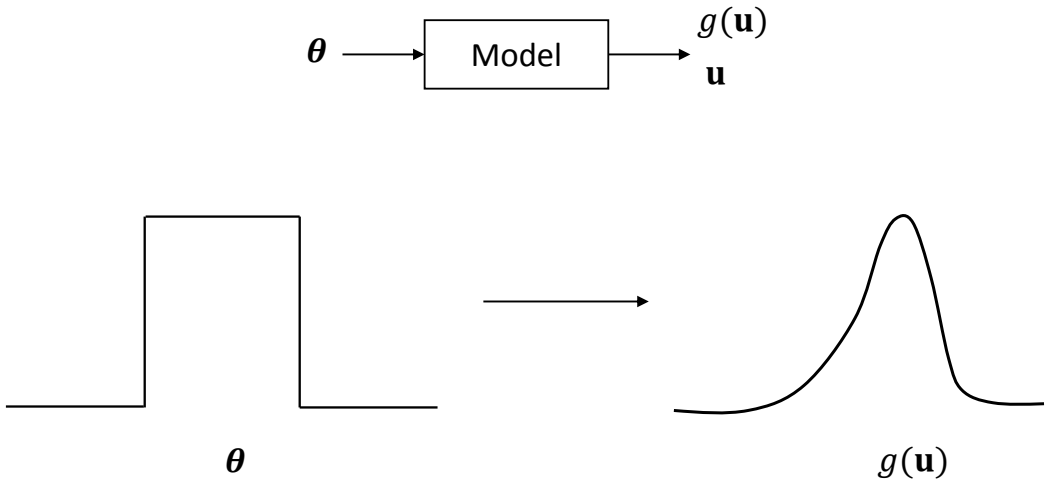
1.1 Motivation

1.1.1 The many-query context

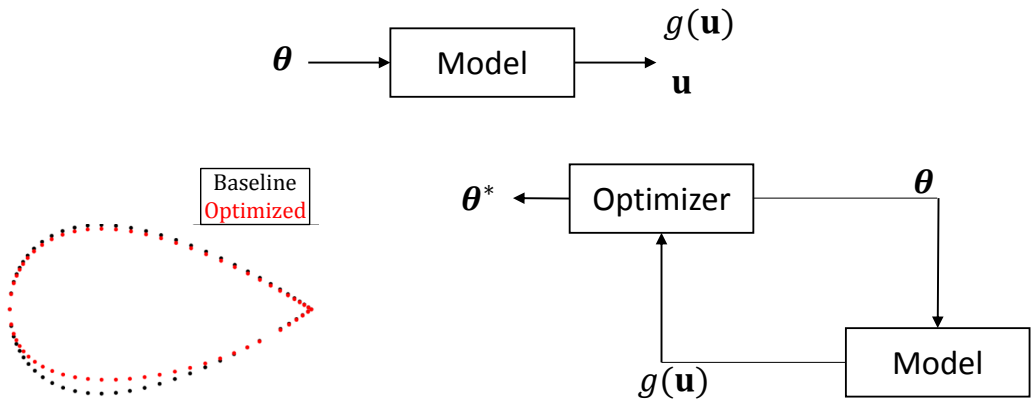
In the design of complex engineered systems specifically for aerospace applications, mathematical models that capture the general physical characteristics of the system are popular. With the advent and constant rise of computing power, such models are preferred to physical testing during the early phases of design, since they are significantly cheaper on a relative sense. However to take full advantage of such models, they have to be used in certain *many-query* context where they have to be evaluated several times; see Figure 1.1. Examples of such many-query context include *design space exploration* where, one might be interested in evaluating the performance of a vehicle under different operating conditions to assess the suitability of a design; *design optimization* where, one is interested in finding the best design based on one or more objective functions under a certain parameter space and similarly *uncertainty quantification* where, probability densities of system outputs are of interest. In each of these contexts, the number of times the model is evaluated could be in $\mathcal{O}(1000)$ if not more. Such a budget of function calls might not be feasible in most cases, despite the state of the art in computing resources. In such situations, a certain approximation of the model replaces the actual model. *We are specifically interested in approximations that trade relatively much less accuracy for a significantly high gain in computational efficiency*.



Design Space Exploration



Uncertainty Quantification



Design Optimization

Figure 1.1: A few examples of the *many-query* context in aerospace design. The focus of the thesis is design optimization.

1.1.2 Approximation models in aerospace design

Aerospace design optimization typically involves a high dimensional design space with multiple local optima and/or noisy design variables. To ensure that the globally optimum design is obtained, in addition to having robust and accurate optimization methods, it is also important to use accurate high-fidelity models to search the design space. The search for global optimum using high-fidelity models in engineering design optimization is typically done using Surrogate Based Optimization (SBO) [1, 2, 3]. Surrogate models or Metamodels are developed by regressing a training data set generated via high fidelity models at a pre-determined set of points and the optimum is searched within the surrogate model. There is some sophistication available to this approach where the surrogate model can be adaptively developed such that more samples are concentrated in regions where the optimum is likely to be found [1]. Such models are generally developed using data-fit approaches using *parametric models* such as Response Surface Methodology (RSM) [4], Artificial Neural Networks (ANN) [5] or *non-parametric models* such as Kriging [6] and Radial Basis Functions [7]. While they are easy to implement and assume simple functional forms that are computationally cheap to evaluate, the limitations are that such models (i) suffer from the curse of dimensionality (ii) do not account for the physics of the problem and hence are not guaranteed to provide a certain definite level of accuracy [8] and, (iii) are not amenable to computing the field variables [1]. Another class of surrogate models that differ from the data-fit models by being customizable to the physics of the high-fidelity models are called *physics-based metamodels*. Their name is due to the fact that their formulation ensures they satisfy the governing equations of the system [8]. Specifically, we refer to a class of models that capture the principal characteristics of a high-fidelity model (small and large-scale non-linear phenomena) while only solving a *smaller* version of it, leading to significantly improved computational efficiency while still offering similar accuracy. In the design of

[1] By *field* variables, we mean vector-valued variables that typically represent a scalar quantity (such as pressure or temperature at a spatially distributed grid

emerging aerospace technologies where the physics of system is complex (such as those governed by a non-linear, coupled set of PDEs) and characterized by a high-dimensional input space, such a model when carefully constructed, can offer superior accuracy within known error bounds (see for instance [9, 10, 11]) and the computational efficiency comparable to data-fit surrogate models.

Here, we take some care to distinguish the physics-based metamodels mentioned above to *low-fidelity* model. A low-fidelity model on the other hand is indeed physics-based, but satisfies some *assumption-based* simplified form of a high-fidelity model. For instance, the panel method [12] or the vortex-lattice method [13] assume inviscid and irrotational flow, which simplifies the solution approach to the governing equations but their validity is questionable when the true flow-field might not necessarily be restricted to the inherent assumptions. Often, such models neglect non-linear and small-scale phenomena in the original system, as part of their simplification. While such models are still very useful within the limits of their validity, they might also not offer the computational efficiency of a typical surrogate model and therefore might not be suitable for a many-query situation where there could be $\mathcal{O}(1000)$ function calls to the physics-based model. Often times, a hierarchy of models is used which includes a high-fidelity (expensive) model and one or more low-fidelity (cheap) models, and by applying domain knowledge the expensive models are used only in regions of the design space where the cheaper models are unable to capture the physics accurately enough. This is called the *variable fidelity* [14, 15, 16] approach and is not addressed in this thesis since we assume here that only the high-fidelity model is available to which we are interested in developing a physics-based approximation.

Therefore there is a need to develop models that do capture the underlying characteristics of the true physics of the system while still being computationally cheap. In this thesis, the focus is on physics-based models, that are suitable for real time analysis and many-query problems such as design optimization, uncertainty quantification and optimal control.

Main motivation of thesis

The motivation for pursuing physics-based surrogate models in this thesis are due to 2 primary reasons (i) the capability to naturally approximate **field variables** without any additional computation and (ii) the physics-based nature that **captures the underlying physical relationships** between inputs and outputs accurately but at a fraction of the computational cost. Therefore, if successful, this method would enable a computationally cheap design-space exploration that facilitates reliable decision making during early phases of design by understanding system-level trade-offs. Such real-time analysis and reliable decision-making under uncertainty forms the overarching objective we wish to accomplish via the proposed approach.

1.2 Reduced Order Modeling

Reduced Order Models (ROMs) are a class of physics-based metamodels that develop a low-dimensional representation of the high-fidelity model, the Full Order Model (FOM). They fall under the general category called Reduced Basis Methods[2], since they use a linearly independent set of basis vectors upon which the FOM is projected to give a reduced dimensional model, which can be solved directly for the state variables. The spatial and temporal variation of the state variables of the system can be obtained by solving a much smaller version of the FOM which is computationally cheaper, while still satisfying the governing equations.

The fundamental assumption behind ROMs is that most of the variance of a high-dimensional flow field is explained by a low-dimensional vector space spanned by a set of orthonormal basis vectors. Therefore, by projecting the original governing equations

[2]However in the literature, reduced basis methods are used in the context of linear problems while reduced order models are used in the context of non-linear problems.

onto the space spanned by the basis set results in dimensionality reduction in terms of the number of unknowns being solved for. To illustrate this, consider the discrete mathematical model of a steady, parametric, PDE-governed system written in the form

$$\mathbf{R}(\mathbf{u}, \boldsymbol{\theta}) = 0 \tag{1.1}$$

where $\mathbf{u} \in \mathbb{R}^N$ is the discretized vector of field or *state* variables, $\boldsymbol{\theta} \in \mathbb{R}^p$ is the vector of parameters or *design* variables, N is the grid size and also the degrees of freedom of the FOM and $\mathbf{R} : \mathbb{R}^N \times \mathbb{R}^p \to \mathbb{R}^N$ is the residual vector. Note that $\mathbf{u}$ implicitly is a function of the design variables. The main ingredient of a reduced basis method is that it assumes the state variable can be expressed as a linear combination of an appropriately chosen finite set of *trial* basis vectors that span the subspace $\mathcal{A}^M \subset \mathbb{R}^N$

$$\mathbf{u} = \tilde{\mathbf{u}}(1)\boldsymbol{\phi}_1 + \cdots + \tilde{\mathbf{u}}(M)\boldsymbol{\phi}_M = \boldsymbol{\Phi}\tilde{\mathbf{u}} \tag{1.2}$$

where, $\boldsymbol{\phi}_i$ are the trial basis vectors, $\tilde{\mathbf{u}}(i)$ are the undetermined coefficients (also known as the reduced state vector) and $M << N$ is the number of basis vectors chosen, which is problem dependent. While there are several choices to select the basis vectors, a common and efficient approach is to choose a *Lagrange Subspace* [8] that is formed by using the exact solutions of the FOM at a set of unique combinations of the design variables. That is,

$$\Phi = \mathtt{span}\left\{\mathbf{u}(\boldsymbol{\theta_1}), \cdots, \mathbf{u}(\boldsymbol{\theta}_M)\right\} \tag{1.3}$$

In forming the Lagrange subspace that is spanned by the basis vectors, linear independence is necessary in order to eliminate redundancy and this is typically achieved via a method such as the Singular Value Decomposition (SVD) [17] or Krylov Subspace Methods [18] or a combination of both [19]. The Proper Orthogonal Decomposition (POD) [20, 21] is one such SVD based method that is commonly used in ROM development and are discussed in section 1.2.2. The actual order reduction of the system comes from the

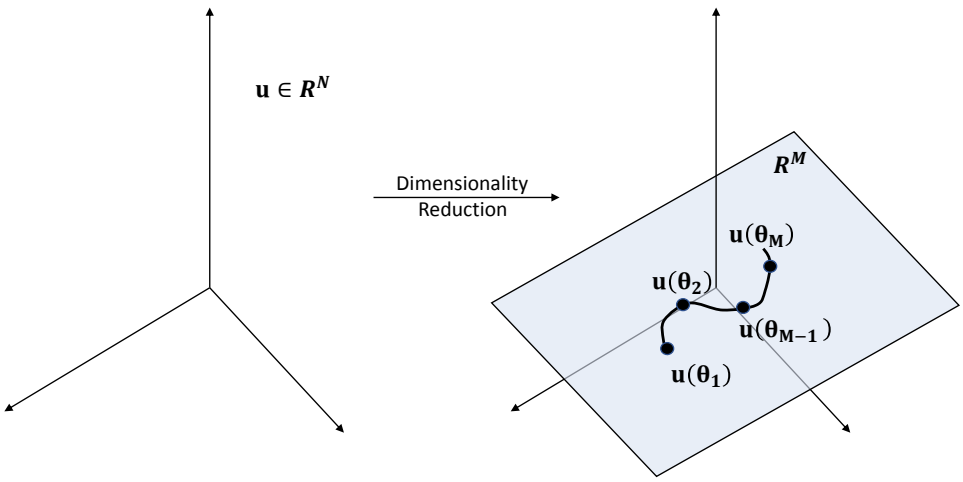


Figure 1.2: Schematic of the dimensionality reduction due to POD. In the left is shown a representation of the original high-dimensional manifold that contains $\mathbf{u}$ and in the right is the low dimensional hyper-plane ($\mathbf{\Phi}$ due to POD), that is induced by the parametric dependence of $\mathbf{u}$. Figure inspired from multiple sources [9, 11]

fact that Equation 1.2 can be substituted into Equation 1.1 such that the true state variable is replaced by its reduced form, which is easier (computationally faster) to solve. In other words, we go from an N-dimensional space to a much smaller M-dimensional space due to the dimensionality reduction. Further, the dimensionality reduction process finds a *linear* embedding of the original non-linear *manifold* [3] that contains the state. This is schematically portrayed in Figure 1.2. Finally, the reduced state vector can be solved for in one of two ways as discussed in the following subsection.

1.2.1 Projection vs Interpolation

The idea behind Equation 1.2 is that if the basis vectors $\mathbf{\Phi}$ can be chosen such that they are globally valid for the design space, then computing $\mathbf{u} \in \mathbb{R}^N$ can be reduced to a problem of computing $\tilde{\mathbf{u}} \in \mathbb{R}^M;\ M << N$. When the numerical approach used to compute $\tilde{\mathbf{u}}$ ensures that the governing equations of the FOM are satisfied, then we call such a model

[3] See chapter 2 for a definition of manifold

physics-based.

The *projection-based* approach to reduced order modeling first substitutes $\mathbf{u} = \mathbf{\Phi}\tilde{\mathbf{u}}$ in the FOM. i.e.

$$\mathbf{R}(\mathbf{\Phi}\tilde{\mathbf{u}}, \theta) = 0 \tag{1.4}$$

Secondly, model reduction may be achieved by projecting the above system onto a suitable *test* basis that span the subspace $\mathcal{B}^M \subset \mathbb{R}^N$ in one of 2 ways which are briefly reviewed in what follows.

Galerkin Projection

The *Bubnov-Galerkin* or *Galerkin* projection [22, 8, 23] assumes that the residual is orthogonal to the chosen approximation subspace, trial basis: $\mathbf{\Phi}$. i.e

$$\mathbf{\Phi}^T\mathbf{R}(\mathbf{\Phi}\tilde{\mathbf{u}}, \boldsymbol{\theta}) = 0 \tag{1.5}$$

Such an approach is an *orthogonal projection* scheme since it enforces orthogonality between the subspace and the residual vector. Note that such an orthogonal projection is essentially similar to the full-rank least squares problem; see for instance [24, 25]. It projects the original $N \times N$ system onto a $M \times M$ system that can be solved directly in the case of linear systems or iteratively (such as the Newton's method) in the case of non-linear systems.

Least Squares Petrov-Galerikin Projection

The *Least Squares Petrov-Galerikin Projection (LSPG)* scheme directly minimizes the L_2 norm of the residual vector with respect to the reduced state variable. i.e.

$$\tilde{\mathbf{u}} = arg\min_{\tilde{\mathbf{u}}} \|\mathbf{R}(\mathbf{\Phi}\tilde{\mathbf{u}}, \boldsymbol{\theta})\|_2 \tag{1.6}$$

When $\mathbf{R}(\mathbf{\Phi}\tilde{\mathbf{u}}, \boldsymbol{\theta})$ represents a non-linear system, the LSPG approach requires an equivalent of the non-linear least-squares minimization [26] to solve for the reduced state variable. As can be directly seen, the LSPG reduces the original problem with N degrees of freedom into the reduced problem of $M << N$ degrees of freedom, where $\tilde{\mathbf{u}}$ is the unknown.

The choice of the appropriate projection scheme depends on the specific problem under consideration. Specifically, the properties of the system matrices that govern the FOM determine the appropriate choice. The Galerkin projection is known to converge as the size of the basis set M increases as long as the system matrix is Hermitian Positive Definite [27]. However, when the system matrix is non-Hermitian, the LSPG is a preferred choice since it applies to generalized matrices [28]. Further, for certain choice of test and trial bases, it can be shown that the LSPG projection on the original governing equations is equivalent to the Galerkin projection on the normal equation (see Chapter 2).

Interpolation

On the other hand, an interpolation approach directly interpolates the reduced state vector, $\tilde{\mathbf{u}}$ in the design variable space. Such an approach does not depend on the FOM and is similar in characteristics to the general data-fit surrogate models that are applicable to black-box systems, and therefore suffer from the same limitations that such models are subjected to, as explained previously. In this thesis we focus solely on projection-based approaches. See [29, 30, 31] for some examples on interpolation based ROM development.

1.2.2 Proper Orthogonal Decomposition

POD is known by different names: *method of empirical orthogonal functions* introduced by Lorenz for the study of weather prediction [32], the *Karhunen-Loeve expansion* in the fields of image compression, data analysis and stochastic processes [33, 34, 35, 36]. POD was introduced in the context of turbulent flow modeling by Lumley [21] and have been explored with the same goal by Sirovich [37, 38, 39], where it was used to characterize the

coherent structures in the flow from wind tunnel measurements. POD has a special characteristic of *optimality* in that it provides the most efficient means to capture the dominant components of a process [40, 41, 42]. Given a state variable $\mathbf{u} \in \mathbb{R}^N$ which may be the numerical solution of a PDE on a computational mesh of size N, the POD expresses $\mathbf{u}$ as the linear combination of a finite number of k basis vectors $\boldsymbol{\phi}_i \in \mathbb{R}^N$. That is,

$$\mathbf{u} \approx \sum_{i=1}^{k} \tilde{\mathbf{u}}(i)\boldsymbol{\phi}_i \tag{1.7}$$

where, $\tilde{\mathbf{u}}(i)$ is the ith component of $\tilde{\mathbf{u}} \in \mathbb{R}^k$ and are the coefficients of the basis expansion. Denoting $\boldsymbol{\Phi}_k = [\boldsymbol{\phi}_1, ..., \boldsymbol{\phi}_k]$, the Equation 1.7 [4] can be written as

$$\mathbf{u} \approx \boldsymbol{\Phi}_k \tilde{\mathbf{u}} \tag{1.8}$$

Therefore, by substituting Equation 1.8 onto the original governing equations, we may solve for $\tilde{\mathbf{u}}$ instead of $\mathbf{u}$, which is more efficient since $k << N$. The POD basis is determined such that it minimizes the error between the state variable and its orthogonal projection onto Φ. Given M snapshots of the state variables $\mathbf{U} = [\mathbf{u}^1, ..., \mathbf{u}^M] \in \mathbb{R}^{N \times M}$ which may be obtained by solving the FOM for various independent parameter combinations, the POD basis minimizes

$$\boldsymbol{\phi}_i := \min_{\boldsymbol{\phi}_i} \sum_{j=1}^{M} \|\mathbf{u}^j - \sum_{i=1}^{k} ({\mathbf{u}^j}^T \boldsymbol{\phi}_i)\boldsymbol{\phi}_i\|_2^2 = \min_{\boldsymbol{\phi}_i} \sum_{j=1}^{M} \|\mathbf{u}^j - (\boldsymbol{\Phi}\boldsymbol{\Phi}^T)\mathbf{u}^j\|_2^2 \tag{1.9}$$

where $(\boldsymbol{\Phi}\boldsymbol{\Phi}^T)\mathbf{u}^j$ is the orthogonal projection of $\mathbf{u}^j$ onto $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi}^T\boldsymbol{\Phi} = \mathbf{I}$. It can be shown that [40, 41] the solution to the above equation is the same as the left singular vectors of the snapshot matrix. That is,

$$\mathbf{U} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{W}^T \tag{1.10}$$

[4] Note that '$\approx$' symbol is introduced here since the M (in Equation 1.2) is replaced with k, $k < M$ indicating that the trial basis set is truncated after k columns

then $\mathbf{\Phi}_k$ represents the first k columns of $\mathbf{V} \in \mathbb{R}^{N \times N}$. The L_2 error in approximation of the state variables due to the POD basis expansion is then given as

$$\sum_{j=1}^{M} \left\| \mathbf{u}^j - (\mathbf{\Phi}\mathbf{\Phi}^T)\mathbf{u}^j \right\|_2^2 = \sum_{i=k+1}^{M} \sigma_i^2 \tag{1.11}$$

where σ_i are the singular values corresponding to the ith column of $\mathbf{V}$ and are also the ith diagonal element of $\mathbf{\Sigma}$. The POD step is followed by the projection step where the FOM governing equations are projected onto the reduced set of POD basis thereby reducing the original $N \times N$ system into a $k \times k$ system where k is the dimension of the reduced POD basis set. This is briefly discussed in the following section.

1.2.3 Projection Step

The projection step projects the original FOM onto the low-dimensional subspace spanned by the POD basis vectors, forming the ROM [40]. Consider the discrete representation of the FOM presented originally in Equation 1.1 in its expanded form

$$\mathbf{A}(\theta)\mathbf{u} = \mathbf{f}(\mathbf{u}) \tag{1.12}$$

where $\mathbf{A}(\boldsymbol{\theta}) \in \mathbb{R}^{N \times N}$ is the linear differential operator that arises due to the discretization of linear terms, $\boldsymbol{\theta} \in \mathbb{R}^p$ is a vector of design parameters and $\mathbf{f}(\mathbf{u}) \in \mathbb{R}^{N \times 1}$ is the non-linear operator that arises due to the discretization of the non-linear terms and also lumps the boundary condition discretization terms and source terms if present. Note that in the above equation $\mathbf{R} = \mathbf{A}(\boldsymbol{\theta})\mathbf{u} - \mathbf{f}(\mathbf{u})$. This represents the full-order system with N unknowns. The projection step now is carried out using either the Galerkin or LSPG projections as explained early on. Therefore for an appropriately chosen test basis, $\mathbf{\Psi}_k$ the projection step enforces orthogonality between the POD basis and the residual [5]. Removing

[5]Note that $\mathbf{\Psi}_k = \mathbf{\Phi}_k$ leads to the Galerkin projection while $\mathbf{\Psi}_k = \mathbf{A}\mathbf{\Phi}_k$ leads to the LSPG (which is shown in chapter 2.

the $\boldsymbol{\theta}$ for convenience of notation, this is equivalent to

$$\boldsymbol{\Psi}_k^T(\mathbf{A}\mathbf{u} - \mathbf{f}(\mathbf{u})) = 0 \tag{1.13}$$

Since the reduced state variable $\tilde{\mathbf{u}} \approx \boldsymbol{\Phi}_k^T\mathbf{u}$, the above equation can be written as

$$\boldsymbol{\Psi}_k^T\mathbf{A}\boldsymbol{\Phi}_k\tilde{\mathbf{u}} = \boldsymbol{\Phi}_k^T\mathbf{f}(\boldsymbol{\Phi}_k\tilde{\mathbf{u}}) \tag{1.14}$$

defining the reduced matrix $\tilde{\mathbf{A}} = \boldsymbol{\Psi}_k^T\mathbf{A}\boldsymbol{\Phi}_k \in \mathbb{R}^{k\times k}$,

$$\tilde{\mathbf{A}}\tilde{\mathbf{u}} = \boldsymbol{\Psi}_k^T\mathbf{f}(\boldsymbol{\Phi}_k\tilde{\mathbf{u}}) \tag{1.15}$$

The above equation represents a reduced system with $k << N$ unknowns that can be solved efficiently.

1.2.4 Non-Linearity Treatment

The computation of the non-linear term (RHS in Equation 1.15) still involves operations in $\mathcal{O}(N)$ (due to the product $\boldsymbol{\Phi}_k\tilde{\mathbf{u}}$) and renders the computation inefficient since it has to be evaluated repeatedly in an iterative procedure such as the Newton's method. However, this can be overcome with methods that involve either a local linearization [43] or subspace interpolation methods such as Gappy-POD [35, 44, 45], Best Points Interpolation Method (BPIM) [46] or the Discrete Empirical Interpolation Method (DEIM) [47]. The subspace interpolation methods (BPIM and DEIM) expand the non-linear terms as a linear basis expansion and with undetermined coefficients. The coefficients are then determined by computing the non-linear term only at a small subset of "best" or "optimal" points; the individual methods essentially differ only in the criteria used to select the points. Since the subspace dimensionality of the non-linear terms is typically significantly smaller than the model dimensionality (N), it achieves significant computational efficiency. Such methods are specifically attractive for model-reduction due to its similarity in operation to the

POD-based ROM development itself; essentially with an offline phase of collecting snapshots and determining orthonormal bases and an online phase of coefficient determination. Secondly from an algorithmic point of view, their implementation blends in with the ROM development without requiring detailed detours. In this thesis, we adopt the DEIM due to its success in non-linear model reduction [48, 29] and its suitability to generalized non-linear terms [49].

1.3 Literature Review

Before stating the objectives of this thesis, we briefly review the relevant literature.

1.3.1 General Applications

Model reduction for non-linear parametric systems is specifically suited for design optimization and has been applied to a variety of aerospace design problems in the past decade including aerodynamic inverse design and missing data re-construction [44, 50], probabilistic aerodynamic analysis [51], aero-elastic applications [52, 53, 54, 51, 55, 56, 57, 58, 59, 60], aero-thermo-elastic applications [61, 62], reacting flow modeling [63], hypersonic thermal protection system design [64, 48], turbomachinery [65, 66, 67], rotary wing and prop-rotor aerodynamics [68, 69, 70] and supersonic flow modeling [71, 72, 73, 74]. In almost every instance, the ROM method was an intrusive projection-based approach or a non-intrusive interpolation based approach. We intend to develop a non-intrusive projection-based approach in this thesis.

1.3.2 Shape Optimization

The core application area in this thesis is Aerodynamic Shape Optimization (ASO) which is a PDE-constrained optimization (see [75] for a quick introduction and associated challenges). In such problems, the governing equations of the FOM are constraints to be satisfied (in addition to other constraints) which requires a high-fidelity function call each

time the constraint is evaluated. A common approach to solving this problem is to use adjoint-based sensitivities [76, 77, 78, 79] of the objective function in a gradient-based optimization setting. However when a gradient-free optimization approach is of interest in order to determine the globally optimal solution, using the FOM in-the-loop is not feasible. Typically, data-fit surrogate models are used in such contexts [80, 3, 81, 82, 83, 84, 85]. A well-constructed ROM finds a natural application here due to its capability of offering comparable accuracy to the FOM while being significantly cheap computationally. The idea of using ROMs as surrogates in a PDE-constrained optimization setting has been attempted in the past. For instance, Amsallem et al [86] evaluate the approach on a nozzle shape optimization problem. LeGresley [50] and Bui-Thanh [44] demonstrated its application for inverse-airfoil design and similar work has been done in [87, 88, 89, 27, 90]. Again, we aim to demonstrate it with a projection-based ROM developed using black-box models and use it with a gradient-free optimizer, which has not been done as of writing this document to the best of our knowledge.

1.3.3 Non-Intrusive ROM

The current state of the art of ROM development is *intrusive* in nature since it requires access to the governing equations of the FOM to construct the ROM. However, when the FOM is solved using a black-box code such as commercial Computational Fluid Dynamics (CFD) packages, the governing equations (in discrete form, as necessary for projection-based approaches) are inaccessible, posing a hurdle to the ROM development.

This opens up a very fundamental question about the feasibility of reduced order modeling, given a black-box code. This question is important because the need for commercialization and protection of intellectual property has lead to high-fidelity models being widely available as a black-box where the user only has control over passing inputs and parsing the output. Therefore, if feasible, then reduced order modeling is expected to broaden its scope significantly. This thesis attempts to answer this question primarily.

It should be emphasized that there have been attempts in the past towards a non-intrusive approach to ROM. The most commonly found approach to the author was to use some-form of interpolation technique to determine the reduced state vector (see section 1.2.1). RBF is particularly suitable due to their linear structure and their ability to handle scattered high-dimensional data; see [91, 31, 30] for examples of the use of RBF in reduced order modeling. A data-fit surrogate model, such as polynomial response surface or artificial neural nets can be used in equal effect as the RBFs. However, interpolating the reduced state vector poses two challenges: (i) the interpolated value of (reduced) state has no guarantee to still satisfy the governing equations. This is because the actual variation of the state in the parameter space is unknown and highly problem dependent and hence might not be captured by a generic function, and (ii) an adequately accurate data-fit surrogate model requires a sufficiently rich training dataset. This means that the parameter space has to be sampled densely which suffers from the *curse of dimensionality* in high-dimensional problem. In such cases the cost of constructing an accurate ROM could outweigh the benefits associated with using them in a design optimization setting. While, there are advantages to this approach such as ease-of-use and circumventing the model stability issues, the author considers them in essence to be another data-fit surrogate model which inherits their associated limitations. On the other hand, the projection-based approach to model reduction ensures that certain reduced form of the FOM always does satisfy the governing equations. While they do indirectly suffer from the curse of dimensionality, their *physics-based* characteristic enables development of models with sound theoretical guarantees on accuracy [10, 92] which is attractive from the perspective of using them in the design of advanced next-generation aerospace concepts. Therefore, we revisit the original question posed at the beginning of this subsection to ask if a ***projection-based reduced order modeling approach is feasible with black-box codes?***.

When it comes to projection-based model reduction, access to the governing equations (system matrices, Eq. 1.12) is necessary. Therefore in the case of black-box models,

these matrices should be inferred/approximated in some form. The recent emergence of *equation-free* methods which infer the governing equations in a data-driven manner, such as in [93], has shown that with *machine learning* tools and available domain knowledge, the governing model can be constructed from the outputs it generate. Such an approach works very well to infer linear systems and non-linear systems with low-order polynomial-type non-linearities [94]. Further, there has been a recent revival of the *Koopman theory* [95], which shows that even for non-linear dynamical system (in the state space), a linear operator on a higher-dimensional *observable* space can be constructed with data (in the form of observable snapshots). Such an approach then provides with system matrices which can then be considered the governing equations upon which model reduction can be performed [96, 97][6]. However, such approach depends on trajectory data, i.e. snapshots of the state/observables in time (in addition to parameter space). This thesis proposes a method to overcome this limitation.

This thesis proposes a method that enables projection-based construction of ROMs with black-box full models for steady, non-linear parametric systems with generalized non-linearities. It specifically addresses the situation where there is knowledge of the governing equations (*domain knowledge* [7]), however there is no access to the source code that implements the numerical solution, which is typically required to perform model reduction. It first represents the governing equation in terms of a set of *observables*, drawing from the Koopman theory of partial differential equations [96]. Such a representation transforms the governing equation into a higher dimensional but linear system, where the transformed linear terms are acted upon by a linear operator. Secondly, the linear operator matrix $\mathbf{A}$ is approximated via a direct discretization of the transformed equation. The method recognizes that $\mathbf{A}$ is the discrete version of a few standard linear differential operators such as

[6] We discuss the Koopman theory and associated methods in a bit more detail in Chapter 2.

[7] Here by domain knowledge we mean that the governing equations in continuous PDE form is known. If not, there is atleast partial knowledge about what linear and non-linear terms exist and the associated differential operators that act on them. This would be more clear through the general formulation in Chapter 2 and specific examples in Chapters 3 and 4

the gradient, divergence, laplacian and curl. Additionally, $\mathbf{A}$ requires only the computational grid and the parameters of the system which are always known. For a given snapshot of the FOM (in terms of the observables) and the matrix $\mathbf{A}$, the right-hand side, $\mathbf{f}$ of the governing equations (which lumps the boundary condition and source terms) is recovered. The overall method is summarized in the Figure 1.3. Note that the $\mathbf{f}$ now is independent of the state since all the non-linear terms are cast in terms of an observable and hence is suitable for a direct element-wise interpolation for parameter changes. This allows one to generate the system matrices $\mathbf{A}$ and $\mathbf{f}$ that are necessary for projection-based reduced order modeling.

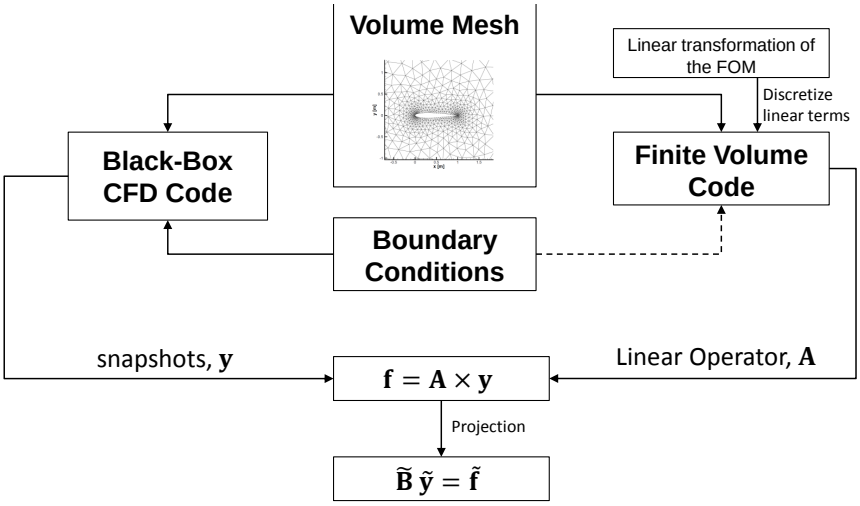


Figure 1.3: Flowchart of overall workflow. Note that certain *closure* equations are required to augment the ROM as explained in Chapter 2

A note on black-box models

In the context of this thesis, the black-box assumption of the high-fidelity model means that beyond passing inputs and reading outputs, there is no additional control over the high-fidelity model. Specifically, there is no access to the discrete form of the governing equations necessary to perform projection-based model reduction. Further, we assume that there are no Application Programmer Interface (API) available with the black-box code which would facilitate access to the discrete form of governing equations or the associated residuals. This assumption is specifically valid with the black-box code (STARCCM+) used in this work at the time of writing this thesis. Under such circumstances and with knowledge of the governing equations in continuous PDE form, we show that the proposed methodology enables projection-based model reduction for static, parametric non-linear systems available as a black-box.

1.4 Thesis Objectives

The main objectives of this thesis are the following

1. To devise a methodology that enables projection-based reduced order modeling of PDE-governed high-fidelity models available as a black-box.
2. To demonstrate the capability of the ROM to accurately predict the state space and outputs of non-linear static parametric systems
3. To demonstrate the capability of the ROM to be used in many-query contexts such as global optimization and uncertainty quantification

1.5 Thesis Contributions

The following are the main contributions of this thesis

1. Development of a framework that achieves objective-1. Such a framework is customizable to any PDE-governed model available as black-box, given certain requirements as discussed in Chapter 2
2. Demonstrate that the method can be constructed at a computational cost that scales linearly with grid size N which is as good as other methods that exist in literature to address similar problems
3. Perform extensive validation studies under different parameter settings and flow regimes and identify model hyper parameters that are customizable for improved accuracy
4. Demonstrate that the method offers orders of magnitude in computational cost-improvement making it suitable for many-query, real-time decision making scenarios

1.6 Thesis outline

The rest of the thesis is organized as follows. In Chapter 2, the details of each aspect of the methodology are provided. The chapter concludes with a chart of the overall framework and the associated algorithm. In Chapter 3, feasibility test of the method is performed using canonical PDE test cases. Chapter 4 performs detailed model validation under transonic and subsonic flow conditions with two types of parameters namely: (i) flow paramters and (ii) shape parameters. Strengths and weaknesses of the approach are demonstrated and model parameters that affect accuracy are identified. In Chapter 5, the method is successfully demonstrated on a few application problems involving design optimization and uncertainty quantification. Chapter 6 provides concluding remarks and some directions into future work.

CHAPTER 2
METHODOLOGY

The overall methodology developed in this thesis draws from several fields within applied mathematics. Particularly, the following form the main building-blocks namely, *the Koopman theory*, *finite volume method* for discretization of PDEs, *projection-based model order reduction* and finally *interpolation* of parametric matrices *in tangent spaces to differential manifolds*. The Koopman theory forms the basis of the present approach where a linear representation of the non-linear system is obtained. The finite volume method is the tool used to discretize linear differential terms as they are well suited for unstructured computational grids and are almost the standard in commercial CFD applications. The ROM interpolation in this thesis draws from differential geometry in order to address the manifold-embedding of the ROM system matrices. Such an approach is essential in ensuring the fundamental properties of the system matrices are retained post-interpolation.

The proposed methodology is divided into 5 sequential steps as outlined in Figure 2.1. As explained in the figure, the first 2 steps are the novel contributions of the proposed approach which allows one to extract the governing equations of the non-linear static PDE system. The projection step is typical of any model reduction method which achieves the state-space dimensionality reduction via the POD as described in Section 1.2.2. The interpolation step provides an efficient alternative to reconstructing the ROM for every new parameter instance. Finally, the resulting ROM is solved via a suitable numerical method since they represent a reduced version of the original non-linear static system.

This chapter describes each of the aforementioned topics in necessary detail while further detail is presented in the Appendices when appropriate. The chapter concludes with an overview of the computational complexity (offline cost) for the overall model development to emphasize how it shall reap the benefits of computational cost when used in a

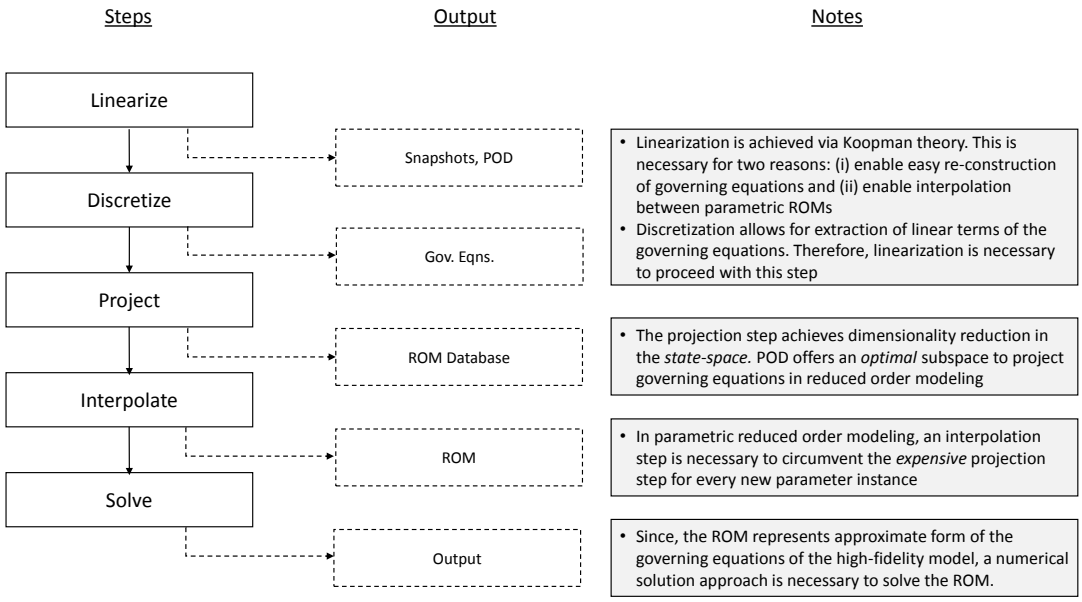


Figure 2.1: Outline of the overall methodology. The *Linearize* and *Discretize* steps achieve the discovery of the governing equations which are unique to the proposed approach in this thesis to overcome the limitations of using a black-box for the high-fidelity model. The remaining steps are typical of any projection-based model reduction method

many-query context. A summarizing flowchart of the methodology is presented at the very end.

2.1 Koopman Theory

2.1.1 Koopman theory for Non-Linear Dynamic Systems

The Koopman Operator theory was originally postulated by Bernard Koopman in 1931 [95]. In a nutshell, this theory suggests that a linear representation of a non-linear dynamical system is possible by appropriately transforming the state variables; however the transformed linear system is infinite dimensional. In certain special cases, it has been shown to be possible to extract a closed-form finite-dimensional linear system [98, 99], but this does not generalize well to all non-linear systems. When such finite-dimensional linear transformation is not possible, data-driven approaches via the Dynamic Mode Decomposition (DMD) [100, 101] are used to get an approximation (such as in the least-squares sense)

to such a linear operator. Recently, [102] has combined the Koopman and DMD to apply them towards non-linear model order reduction. In this thesis, the Koopman theory is of interest specifically due to the linear representation it provides. However, rather than using the DMD, we use the under-determined linear representation of the Koopman operator in combination with a set of algebraic equations in the form of non-linear constraints to provide closure.

We begin by defining the Koopman theory stated as follows

Definition 1. *(Koopman Theory [95])*

For a dynamical system

$$\frac{d\mathbf{u}}{dt} = \mathbf{N}(\mathbf{u}) \tag{2.1}$$

where $\mathbf{u} \in \mathbb{R}^N$ *is the state variable and* $\mathbf{N}$ *is a non-linear operator. As it is,* $\mathbf{u}$ *is in a non-linear manifold,* $\mathcal{M}$*. The Koopman operator* $\mathcal{K}$ *acts on a set of scalar observable variables,* g *which comprise the vector* $\mathbf{g}$ *such that* $\mathbf{g} : \mathcal{M} \to \mathbb{R}$ *so that*

$$\mathcal{K}\mathbf{g}(\mathbf{u}) = \mathbf{g}(\mathbf{N}(\mathbf{u})) \tag{2.2}$$

Essentially Definition 1 means that, the original system defined by the state variable $\mathbf{u}$ which is inherently in a non-linear manifold, can be represented as a linear system when defined in terms of the observables. The Koopman operator $\mathcal{K}$ offers this mapping between the non-linear manifold and the linear space. This way, the Koopman operator defines an infinite dimensional but linear operator that still describes the dynamics of the original system. That is, given the following evolution of the state for a non-linear dynamical system

with the evolution map, $\mathbf{F}$ and with t as the time parameter

$$\mathbf{u}_{t+1} = \mathbf{F}(\mathbf{u}_t) \tag{2.3}$$

The Koopman operator is defined as

$$\mathcal{K}\mathbf{g} = \mathbf{g} \circ \mathbf{F} \tag{2.4}$$

where $\circ$ represents the composition operator. This means that,

$$\mathcal{K}g(\mathbf{u}_t) = g(\mathbf{F}(\mathbf{u}_t)) = g(\mathbf{u}_{t+1}) \tag{2.5}$$

which gives the linear evolution

$$\mathbf{g}_{t+1} = \mathcal{K}\mathbf{g}_t \tag{2.6}$$

Therefore, an infinite dimensional but linear map, $\mathcal{K}$ exists for a non-linear system. Further the recently developed Koopman Theory of PDEs [96] states that upon judicious choice of a set of scalar observables that are functions of the state variables, the original non-linear dynamical system can be exactly represented by some higher dimensional but linear system. Such a closed-form finite dimensional linear representation of a non-linear system is not always possible as mentioned earlier. However, it has been shown [97] that a finite dimensional approximation, K to $\mathcal{K}$ is possible. Such an approach takes a *machine learning* approach by considering a large dictionary of functions of the state variables as candidate observables (typically chosen based on domain knowledge) and finds the best set from data (snapshots) in the least-squares sense. Additionally, [94] have shown that such a method converges to the actual governing equations (in terms of the linear and non-linear operators) with sufficient trajectory data.

2.1.2 Koopman theory for Non-Linear Static systems

This work draws from the Koopman theory only in the sense that it represents the underlying steady non-linear parametric system as a linear system in terms of the observables. In this regard, the present method assumes knowledge of the governing equations[1] of the FOM, and each non-linear term and input is assigned to a unique observable. This leads to a transformed system in terms of observables which typically out-number the original system in terms of the state variables. Also, since the present work focuses on static parametric systems, there is no trajectory data and hence an approximation to the Koopman operator as in the Extended-DMD [97] is not possible. This leaves with a linear, underdetermined system, which needs to be closed with additional equations as will be shown in what follows.

Consider a static non-linear system of the form

$$\mathbf{N}(\mathbf{u}) = 0 \tag{2.7}$$

where $\mathbf{N}$ represents a non-linear operator on the state variable $\mathbf{u}$. Let $g(\mathbf{u})$ represent an observable that is a function of the state variable, $\mathbf{u}$. We state that

$$\mathbf{N}(\mathbf{u}) \rightarrow \mathbf{L}[g(\mathbf{u})] \tag{2.8}$$

where, $\mathbf{L}$ is a linear operator acting on the observables. Discretizing the above equation, we get

$$\mathbf{L}[g(\mathbf{u})] \approx \mathbf{A}g(\mathbf{u}) + \mathbf{b}_a = 0 \tag{2.9}$$

Where, $\mathbf{b_a}$ represents a vector that arises due to the discretization of the boundary con-

[1] Note that the assumed knowledge refers only to the continuous form of the governing equations and not the discretized form which typically require access to the source code of the computational implementation of the mathematical model.

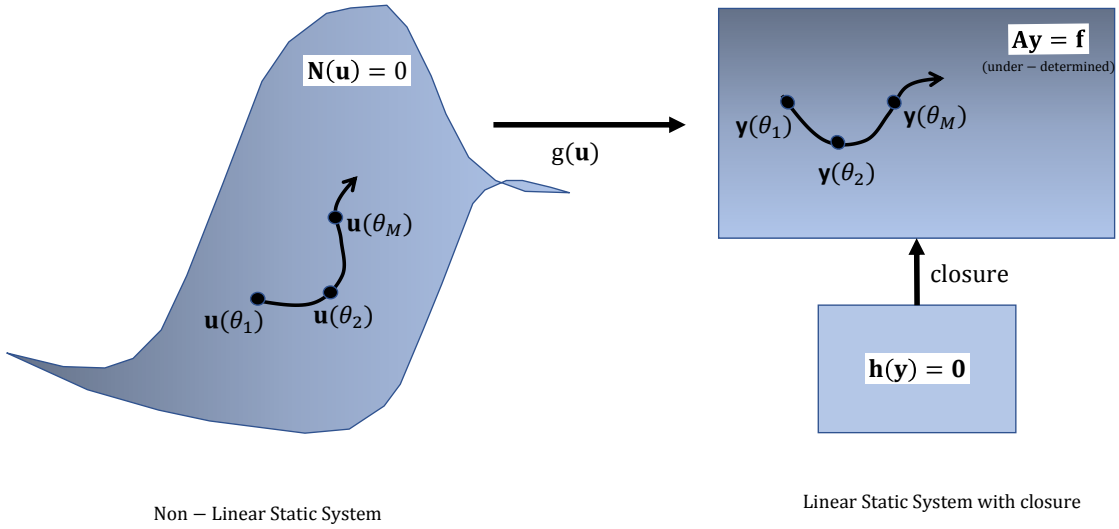


Figure 2.2: Graphical representation of the present methodology. The original non-linear static system is transformed to an under-determined linear system with closure

ditions, lumps the source terms if present and is also the RHS of the FOM. Note again that since each non-linear term is transformed into an observable, the RHS ($-\mathbf{b_a}$ in Equation 2.9) no longer depends on the state $\mathbf{u}$ and hence Equation 2.9 is a linear (but under-determined) system. Setting $g(\mathbf{u}) \rightarrow \mathbf{y}$ and $-\mathbf{b_a} \rightarrow \mathbf{f}$, leads to the transformed linear system

$$\mathbf{A}\mathbf{y} = \mathbf{f} \tag{2.10}$$

We intend to develop the ROM for Equation 2.10 and hence the snapshots are collected in the observable space, $\mathbf{y}$.

In the present work, we differ from the traditional Koopman-DMD for dynamical systems by not using any trajectory data (we address static systems). Secondly, the linear operator of the transformed equation is directly approximated by discretizing the linear dif-

ferential terms through a suitable discretization method such as the finite volume method. This operator, applied to a snapshot of the observables itself gives the RHS of the governing equations that lump the non-linear terms, boundary conditions [2] and source terms if present. This way, the present work obtains a linear representation of the governing equations, which is comparatively easier to work with. However, transforming the state-space to observable-space to obtain a linear system does not necessarily form a closed linear system [99, 96, 98] that is consistent with the original non-linear system. And therefore lastly, this work differs in that the linear system is closed with a set of non-linear constraints that establish consistency between the original state variables and the observables. In essence, the state-observable transformation is the only step that is drawn from the Koopman theory. Finally, the model reduction is performed on the transformed equations (the right hand side of Figure 2.2). This is explained in the following section.

2.1.3 Heuristic to apply Koopman theory for static systems

The practical implementation of the Koopman theory in the context of the method proposed in this thesis is summarized by the following heuristic

1. Identify all *primitive* variables of the system. Note that a primitive variable is independent of any other variable of the system. In a system of S coupled PDEs, there are S primitive variables.

2. Map every *term* in the PDE system that is *a function of one or more primitive variables*, into an *observable*. Note that this includes the primitive variable themselves if they occur stand-alone in the system. In a linear PDE, such a mapping leaves the system unchanged. In a nonlinear system, such a mapping transforms the system to a linear system.

[2] The present method allows an explicit specification of boundary conditions as part of the construction of the linear operator. However, in order to keep the overall method non-intrusive they are lumped to the RHS as mentioned.

3. Re-write the PDE system in terms of the observables.

The steps outline above are illustrated in examples in Chapters 3 and 4.

2.2 Model Order Reduction

We begin by representing the FOM in the transformed observable space as

$$\mathbf{A}\mathbf{y} = \mathbf{f} \tag{2.11}$$

where, $g(\mathbf{u})$ is replaced with $\mathbf{y}$ and $\mathbf{f}$ can have parametric dependence but is independent of the state. The observables $\mathbf{y} = [\mathbf{y}_1, ..., \mathbf{y}_O]^T$, where O is the total number of linear and non-linear terms in the FOM and also the total number of observables. For a FOM that is a system of S coupled PDEs, note that $O \geq S$ always and $O > S$ for a non-linear system. Therefore the observables can be written

$$\mathbf{y} = [\mathbf{y}_1, ..., \mathbf{y}_S, \mathbf{y}_{S+1}, ..., \mathbf{y}_O]^T$$

As evident from the previous equation, a side effect of representing the FOM in terms of observables is that it can increase the dimensionality of the system. In this work, we add algebraic equations that establish the non-linear consistency relationship between certain observables and the rest in order to provide closure to the under-determined system. These constraints are of the form

$$h_i(\mathbf{y}) := \mathbf{y}_{S+i} - f(\mathbf{y}_1, ..., \mathbf{y}_S) = 0, \;\; i = 1, \ldots, O - S \tag{2.12}$$

Equation 2.12 along with the transformed FOM in Equation 2.11 together form a closed system upon which model reduction is performed. Denoting $\Phi_i \in \mathbb{R}^{k_i \times k_i}$ to be the reduced

POD bases extracted from snapshots of $\mathbf{y}_i$, the trial basis matrix is defined as

$$\mathbf{\Phi} = \begin{bmatrix} \Phi_1 & & \\ & \ddots & \\ & & \Phi_O \end{bmatrix} \in \mathbb{R}^{ON \times k} \tag{2.13}$$

where $k = k_1 + \ldots + k_O$ and $\tilde{\mathbf{y}} = \Phi_k^T \mathbf{y}$. Note that the POD basis for each observable as separately determined from their respective snapshot matrix ($\mathbf{Y}_i \in \mathbb{R}^{N \times M}$) that takes the following form

$$\mathbf{Y}_i = \begin{bmatrix} \vdots & \vdots & \vdots & & \vdots \\ \mathbf{y}_i^{(1)} & \mathbf{y}_i^{(2)} & \mathbf{y}_i^{(3)} & \ldots & \mathbf{y}_i^{(M)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \tag{2.14}$$

where $\mathbf{y}_i^{(j)}$ is the jth snapshot of $\mathbf{y}_i$ and the POD basis vectors are obtained from the thin-svd of $\mathbf{Y}_i$. Since the full system is non-square (due to introduction of observables), a suitable choice for the test basis for projection is $\mathbf{\Psi}_k = \mathbf{A}\mathbf{\Phi}_k$. Note that this choice of the test basis is equivalent to a galerkin projection ($\Psi_k = \Phi_k$) on the normal equations. i.e. on $\mathbf{A}^T\mathbf{A}\mathbf{y} = \mathbf{A}^T\mathbf{f}$. Let $\mathbf{B} = \mathbf{A}^T\mathbf{A}$; then the projection leads to

$$\Phi_k^T \mathbf{B} \Phi_k \tilde{\mathbf{y}} = \Phi_k^T \mathbf{A}^T \mathbf{f} \tag{2.15}$$

Setting $\tilde{\mathbf{f}} = \Phi_k^T \mathbf{A}^T \mathbf{f} \in \mathbb{R}^k$ and $\tilde{\mathbf{B}} = \Phi_k^T \mathbf{B} \Phi_k \in \mathbb{R}^{k \times k}$, this leads to the reduced order model

$$\tilde{\mathbf{B}}\tilde{\mathbf{y}} = \tilde{\mathbf{f}} \tag{2.16}$$

The ROM given by Equation 4.6 represents a $k \times k$ system which is rank-deficient since it was obtained through an outer product of two rectangular matrices and is solved along with the constraints presented in Equation 2.12, posed as a constrained optimization

problem as shown below

$$\begin{aligned} \underset{\tilde{\mathbf{y}}}{\text{minimize}} \quad & \frac{1}{2}\|\tilde{\mathbf{B}}\tilde{\mathbf{y}} - \tilde{\mathbf{f}}\|_2^2 \\ \text{s.t.} \quad & h(\mathbf{y}) = 0 \end{aligned} \tag{2.17}$$

where $h(\mathbf{y})$ is a non-linear function (as shown in Equation 2.12) that represents the relationship between observables and is problem dependent. In the subsequent chapters, these constraints are illustrated for specific application problems. The main hypothesis of this work is that the ROM given by Equation 4.7 still approximately satisfies the governing equations and this is verified in the Chapters 3 and 4. The optimization problem in Equation 4.7 needs special treatment to handle the non-linear constraint which still depends on the full state of observables, and is efficiently done using the DEIM which is briefly reviewed in Appendix B but further details can be obtained from [47].

2.3 Finite Volume Method

As previously mentioned, the matrix $\mathbf{A}$ in the present method is directly approximated via discretization of the linear terms. A finite volume based approach [103] is used for the discretization due to its suitability to complex geometries with arbitrarily shaped cells. The final matrix $\mathbf{A}$ itself is composed of several matrices each of which represents the discretized form of the linear differential terms present in the governing equations of the FOM. This discretization step forms the most dominant step (in terms of computational cost) of the model building process with the current approach.

In this work, we take the cell-centered finite volume approach where the dependent variable is stored in the cell centers. In this subsection as an illustration we briefly review the discretization of the (i) 2D diffusion operator $\nabla \cdot (\Gamma \nabla)$ which reduces to the laplacian, ∇^2 when $\Gamma = 1$ and the (ii) 2D gradient operator.

2.3.1 Diffusion Operator

Consider the linear diffusion term is given by

$$\mathcal{L}u = \nabla.(\Gamma\nabla u) \tag{2.18}$$

where the diffusion coefficient $\Gamma = \Gamma(x, y)$ is independent of u. Integrating the above term over a cell volume and applying the Gauss-Divergence theorem we get,

$$\int \nabla.(\Gamma\nabla u)dV = \oint (\Gamma\nabla u).\hat{n}ds \tag{2.19}$$

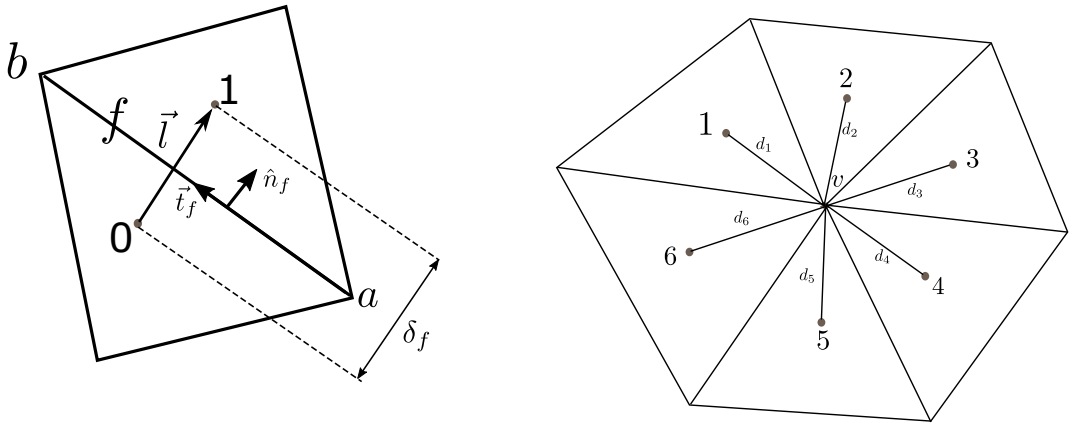
where ds is the face area and $\hat{n}$ is the local surface normal of the face. Assuming that the computational mesh consists of only polygonal faces, each face of a cell has a unique surface normal and hence the above integral can be written as

$$\oint (\Gamma\nabla u).\hat{n}ds = \sum_f \Gamma_f(\nabla u)_f.\hat{n}_f A_f \tag{2.20}$$

where the summation is over the faces of a given cell (subscripted by f), Γ_f is the diffusion coefficient at the interface f and A_f is the face area (length in 2D). The face-center values of diffusion coefficient are obtained from the cell-center values via a linear interpolation (where the interpolants w_f are inverse distance weighted) as

$$\Gamma_f = w_f\Gamma_0 + (1 - w_f)\Gamma_1$$

Now it is a matter of discretizing the RHS of Equation 2.20. Consider two adjacent cells as shown in Figure 2.3a. Here, the center of the cell upon which the discretization is carried out is denoted as '0' and that of its neighboring cell as '1'. The cell centered values of u are denoted as u_0 and u_1 for each of these cells respectively. The vector connecting the cell centers is denoted $\vec{l}$ and that connecting the vertices of the interface f is $\vec{t}_f$ and $\hat{n}_f$ is the unit surface normal. Note that the $\vec{(\,)}$ notation implies vector and the $\hat{(\,)}$ notation implies



(a) Computation of fluxes at cell interface

(b) Distance-weighted interpolation of node values

Figure 2.3: Finite volume cells

a unit vector. Finally, $\delta = \vec{l}.\hat{n}_f$ and since $\hat{n}$ and $\hat{t}$ form an orthogonal set of coordinate vectors, the gradient can be written as

$$(\nabla u) = [\nabla u.\hat{n}]\hat{n} + [\nabla u.\hat{t}]\hat{t} \tag{2.21}$$

At the face f, taking the dot product of $(\nabla u)_f$ with $\vec{l}$ we get

$$(\nabla u)_f.\vec{l} = [\nabla u.\hat{n}_f]\delta + [\nabla u.\hat{t}_f]\hat{t}_f.\vec{l} \tag{2.22}$$

Using taylor series to expand u_0 and u_1 about u_f and subtracting we get

$$u_1 - u_0 \approx \left(\frac{\partial u}{\partial x}\right)_f (x_1 - x_0) + \left(\frac{\partial u}{\partial y}\right)_f (y_1 - y_0) = (\nabla u)_f.\vec{l} \tag{2.23}$$

i.e.

$$(\nabla u)_f.\vec{l} \approx u_1 - u_0 \tag{2.24}$$

In the above equation, the coordinates of the cell centers $'0'$ and $'1'$ are (x_0, y_0) and

(x_1, y_1) respectively. Therefore the term $(\nabla u)_f.\hat{n}_f$ is given as

$$(\nabla u)_f.\hat{n}_f = \frac{u_1 - u_0}{\delta_f} - \frac{[\nabla u)_f.\hat{t}_f]\hat{t}_f.\vec{l}}{\delta_f} \tag{2.25}$$

In the Equation 2.25 the second term is called the 'tangential flux' and denoted as J_T where $\frac{J_T}{\delta_f} = \frac{[(\nabla u)_f.\hat{t}_f]\hat{t}_f\vec{l}}{\delta_f}$. For structured cartesian meshes certain unstructured meshes such as those with equilateral triangular cells, the tangential flux term vanishes and the above expression reduces to standard central differencing. So now we proceed to discretize J_T. We begin by realizing that the Equation 2.24 can be re-written as

$$(\nabla u)_f.\hat{l} \approx \frac{u_1 - u_0}{|\vec{l}|} \tag{2.26}$$

where $|\vec{l}|$ is the inter-cell distance. Similarly, we write

$$(\nabla u)_f.\hat{t}_f \approx \frac{u_a - u_b}{|\vec{t}_f|} \tag{2.27}$$

$|\vec{t}_f|$ is the length of the face connecting vertices a and b in Figure 2.3a. Therefore, the tangential flux J_T can be written as

$$J_T = \left[\frac{u_a - u_b}{|\vec{t}_f|}\right] \hat{t}_f.\vec{l} \tag{2.28}$$

In the above equation, since we know the vectors $\hat{t}_f$ and $\vec{l}$, their dot product is directly obtained. Thus putting it all together into Equation 2.20,

$$\oint (\Gamma \nabla u).\hat{n} ds = \sum_f \Gamma_f \left[\frac{u_{k(f)} - u_0}{\delta_f} - \left[\frac{u_a - u_b}{\delta_f |\vec{t}_f|}\right] \hat{t}_f.\vec{l}\right] A_f \tag{2.29}$$

where, $u_{k(f)}$ represents cell centered values of all the neighbors of cell '0'. The terms (u_a, u_b) in the evaluation of the tangential flux are node-based values (at nodes a and b) and are explicitly treated as the distance-weighted average of all the neighboring cell-centered

values, as shown in Figure 2.3b. i.e. u at a vertex v is given as

$$u_v = \sum_i w_{v,i} u_i \tag{2.30}$$

where $i = 1, 2, 3, ...$ represent the cells surrounding vertex v. The actual number of neighboring cells for a vertex depends on the type of mesh and its location near or away from boundaries. The interpolant $w_{v,i}$ are given by

$$w_{v,i} = \frac{1/d_i}{\sum_i 1/d_i} \tag{2.31}$$

where d_i is the distance of cell center of neighbor cell i to node v.

It should be noted that the faces containing boundary condition information are assumed to be lumped by the RHS vector $\mathbf{f}$. In the finite-volume method, this approach is consistent for a pure Neumann type boundary condition where the flux at the boundary is a known quantity but for other types of boundary conditions this introduces some error in the approximation of the matrix $\mathbf{A}$. However, with knowledge of the boundary conditions, a more accurate approximation of $\mathbf{A}$ can be obtained via the present approach.

2.3.2 Gradient Operator

Consider the gradient operator in 2D given by

$$Gu := \nabla u = \left(\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y} \right) \tag{2.32}$$

It is convenient to denote $\frac{\partial u}{\partial x}$ and $\frac{\partial u}{\partial y}$ by G_x and G_y respectively so that the discretization can be separately developed for each of them. We begin by applying the Gauss-Divergence theorem on every cell to obtain

$$\int \nabla u dV = \oint u.\hat{n} ds \tag{2.33}$$

where u_f is the face-value of the variable u. Just as before, for polygonal cells, the above integral can be approximated using the summation over faces to get

$$\frac{1}{V_0}\oint u_f.\hat{n}ds = \frac{1}{V_0}\sum_f u_f \cdot \hat{n}_f A_f \tag{2.34}$$

Therefore, the two components G_x and G_y can be written separately as

$$G_x \approx \frac{1}{V_0}\sum_f u_{xf}\hat{n}_{xf}A_{xf} \tag{2.35}$$

$$G_y \approx \frac{1}{V_0}\sum_f u_{yf}\hat{n}_{yf}A_{yf} \tag{2.36}$$

where the subscripts $'xf'$ and $'yf'$ denote the x and y components of the variables. Finally, the face-values are approximated from the cell-center values via an inverse-distance weighted interpolant as explained for the diffusion operator. We don't repeat the entire discretization process here since they are almost exactly same as that of the diffusion operator.

2.4 ROM Interpolation

Projection-based model order reduction generally entails an expensive off-line phase of constructing the reduced order system and is also the approach taken in this thesis. However in the case of parametric systems this step has to be repeated for every unique combination of the parameters whose cost might outweigh the benefits that could be potentially reaped from ROMs particularly in many-query and real-time analysis contexts. A natural solution for this problem is to generate a database of ROMs for various snapshots of the parameters and interpolate among the reduced system matrix snapshots to predict the state at new parameter instances. The interpolation of matrices for parameter variations is not trivial since it cannot be done element-wise as if each each element of the matrix was an independent scalar. The main reason is that if the matrix snapshots have special properties

(such as orthogonality, symmetry or positive definiteness) then such an interpolation is not necessarily guaranteed to result in a matrix that retains such properties - thereby modifying the inherent behavior of the linear system to be solved.

A geometrical explanation of the un-suitability of interpolating matrices directly element-wise is that they are embedded in a curved space as opposed to a flat (Euclidean) space and hence a direct interpolation might lead to a new matrix that does not belong to the embedding; see Figure 2.4 and Appendix F for a simple illustration. Intuitively therefore, a certain mapping of the matrices to a locally defined hyperplane is necessary to ensure interpolation is done in a Euclidean space, following which the matrices are mapped back to the manifold. Such curved spaces where the matrices may be embedded are called a *Manifold* (to be defined later in this chapter) and given certain conditions, a *tangent* space to the manifold can be locally defined. Further, mapping between the manifold and its locally defined tangent space are defined via *exponential* and *logarithmic* functions of the matrices. In this section, we provide the necessary preliminaries on *differential* manifolds, *matrix functions* followed by the procedure to interpolate matrices by projecting them to locally defined tangent spaces. As mentioned previously, interpolation forms an integral step in leveraging the full-potential of ROMs in real-time and many-query contexts.

2.4.1 Euclidean Space

The concept of the Euclidean space is to be introduced in order to properly define a manifold. In a very general sense, the Euclidean space is considered a *flat* space where any two geometrical shapes are equivalent if one can be converted to the other through a series of *translations, rotations* and/or *reflections*. Such operations are carried out using vector operations such as addition and inner product. In other words, a Eulcidean n-space is a vector space $\mathbb{R}^n$ with a well-defined inner product [104], such that metrics such as distance and angles can be measured.

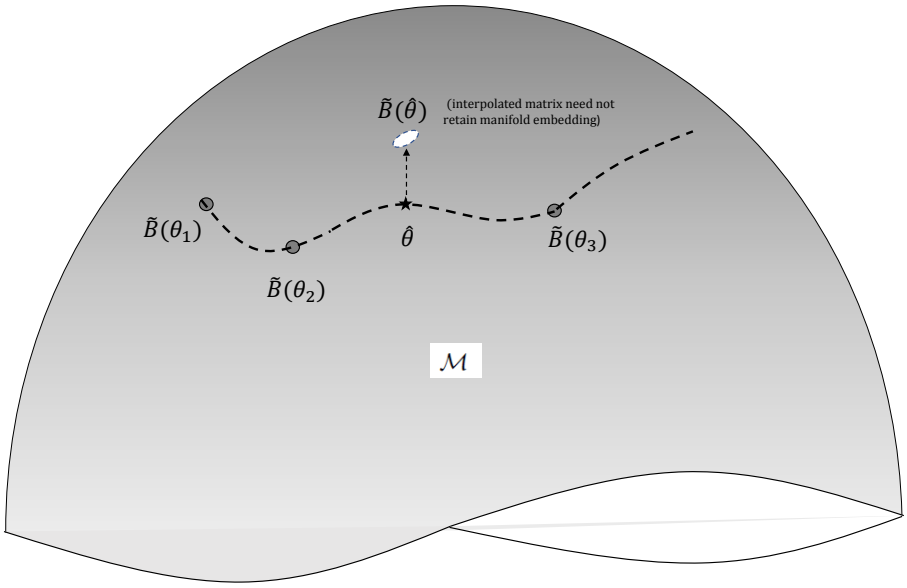


Figure 2.4: A graphical representation of a manifold $\mathcal{M}$ and the embedding of parametric matrices $\tilde{\mathbf{B}}(\boldsymbol{\theta})$. A direct element-wise interpolation of $\tilde{\mathbf{B}}$ at $\hat{\boldsymbol{\theta}}$ may not necessarily result in a matrix $\in \mathcal{M}$

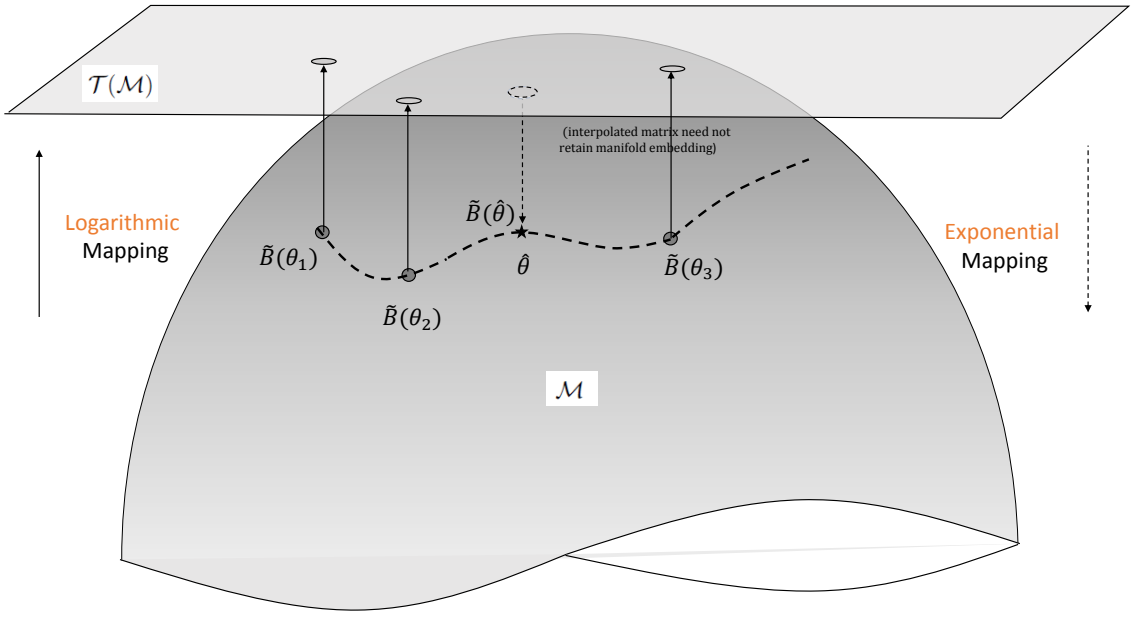


Figure 2.5: A graphical representation of a manifold $\mathcal{M}$ and the associated tangent space. Mapping of matrices to the tangent space precedes interpolation

Definition 2. *(Euclidean n-Space $\mathbb{R}^n$)[105]: is defined as the set of all n-tuples $(x^1, \ldots, x^n)$ of real numbers x^i with element-wise addition and scalar multiplication defined as $x + y = (x^1 + y^1, \ldots, x^n + y^n)$ and $ax = (ax^1, \ldots, ax^n)$ respectively; and a notion of vector lengths defined via the 2-norm $\|x\|_2 = \sqrt{(x^1)^2 + \ldots + (x^n)^2}$*

Within the context of ROM interpolation which involves vector addition and scalar multiplication, it is convenient if the interpolation is carried out on a Euclidean space.

2.4.2 Manifold

A manifold can be thought of as a *space* which locally looks *Euclidean* meaning that every point in a local neighborhood can be mapped 1:1 (also known as *Homeomorphism* [106]) to a flat space. A simple illustration is that the earth has an approximately spherical shape but, locally every surface of the earth looks flat. Therefore the following definition is given to the manifold.

Definition 3. *(Manifold) [106]: A topological n-manifold, $\mathcal{M}$ is locally Euclidean of dimension n if every point $p \in \mathcal{M}$ has a neighborhood U such that there is a homeomorphism ϕ from U onto an open subset of $\mathbb{R}^n$. The pair $(U, \phi : U \to \mathbb{R}^n)$ is called a chart.*

The term *locally homeomorphic* means that each local neighborhood in $\mathcal{M}$ is devoid of any self-intersections and therefore can be mapped to a Euclidean space; see Figure 2.6.

Having defined a manifold, a differential manifold is one in which at every point, there exists a *tangent* space, which is also a vector space. The tangent space has a well defined inner product to quantify the distance and angle between vectors. Such a manifold that has a well defined tangent space at every point (which is a finite dimensional Euclidean space) is

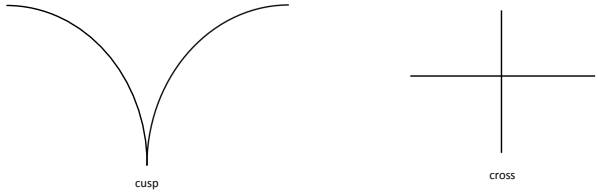


Figure 2.6: (Re-created from [106]) An example of a manifold is the cusp (left) where every neighborhood (including the vertex) can be locally mapped to a Euclidean 1-space. This is not true in the case of the cross (right) where at the intersection a unique mapping to Euclidean space is not possible.

a differential manifold and is called *Riemannian* [107]. Recall that we are interested in the tangent space to a manifold because it is a vector space on which regular interpolation can be performed. However a mapping of points between the tangent space and the manifold (and vice versa) needs to be defined. We begin by defining the tangent space to a differential manifold.

Definition 4. *(Tangent Space): Given a point p in a manifold $\mathcal{M}$, the tangent space to $\mathcal{M}$ at p, $\mathcal{T}_p(\mathcal{M})$ is defined as a vector space in which each vector is a directed line segment anchored at p*

In order to define the mapping between the manifold and its locally tangent space, the following results from differential geometry are useful. A *Geodesic*, $\mathcal{Y}$ is defined as the shortest path between 2 points on a differential manifold [108, 104]. It is also known to be a trajectory ($\mathcal{Y}(t)$) associated with a 2^{nd} order ODE, and is uniquely defined by an initial position $\mathcal{Y}(0)$ and initial derivative, $\dot{\mathcal{Y}}(0)$. With $\mathcal{Y}(1)$ defined as the final point of the geodesic, the following result is presented below

Proposition 1. *(Geodesic Path): The final point of a geodesic is related to the initial point via the following exponential relationship*

$$Exp_{\mathcal{Y}(0)}\dot{\mathcal{Y}}(0) = \mathcal{Y}(1)$$

What Proposition 1 states is that the location of some point on the manifold within a certain neighborhood, is related to another point via the geodesic connecting the two points. Further, it is related to the tangent (first derivative) of the initial point of the geodesic via an exponential relationship. Therefore, by fixing a certain anchor point on the manifold (the initial point of the geodesic) other points within a certain neighborhood can be related to the common anchor point via this relationship. Further, these points on the manifold connected via their respective geodesics to the anchor point can be projected to the tangent plane defined at the anchor point via a *logarithmic mapping*. As a natural consequence of this result, the inverse mapping (tangent plane to manifold) is one that is exponential. Such a mapping acts on the matrices themselves, and hence the concept of *functions of a matrix* are now introduced.

2.4.3 Functions of Matrices

Matrix functions are non-trivial in the sense that they are not a direct extension of scalar functions - for instance, the exponential of a matrix is not the same as the element-wise exponential of the matrix [109]. In what follows, we define three important matrix functions that are relevant to this thesis namely (i) the matrix exponential, (ii) the matrix logarithm and the (iii) matrix square root.

Exponential Mapping

As stated before, the exponential mapping maps points on the tangent space to a manifold to the manifold itself. The exponential of a matrix, $e^{\mathbf{A}}$ is defined for any non-singular $\mathbf{A} \in \mathbb{R}^{n \times n}$ as follows

Definition 5. *Matrix Exponential:* $e^{\mathbf{A}}$

$$e^{\mathbf{A}} = \sum_{k=0}^{\infty} \frac{1}{k!} \mathbf{A}^k \tag{2.37}$$

where $\mathbf{A}^0 = \mathbf{I}$.

Such a series is known to converge as long as $\mathbf{A}$ is non-singular. In terms of computing the logarithm itself, we use the scaling and squaring method with Pade approximant given by Higham [109].

Logarithmic Mapping

The logarithm of a matrix ($\mathbf{B} = Log\mathbf{A}$) is the inverse transform of the exponential and satisfies the definition $\mathbf{A} = e^{\mathbf{B}}$. For a square matrix $\mathbf{A}$, the inverse is defined as

Definition 6. *Matrix Logarithm:* $Log\mathbf{A}$

$$Log\mathbf{A} = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{(\mathbf{A} - \mathbf{I})^k}{k} \tag{2.38}$$

Unlike, the matrix exponential, the logarithm of a matrix (i) is non-unique (ii) does not always exist and (iii) is not guaranteed to converge always. However when $\mathbf{B}$ is invert-

ible, $\|\mathbf{B} - \mathbf{I}\| < 1$ and the eigenvalues of $\mathbf{B}$ do not lie on $\mathbb{R}^-$ then the infinite series in Equation 2.38 is guaranteed to converge to the unique *principal logarithm* [109, 110, 111].

Square Root of a Matrix

The square root of a matrix $\mathbf{A}$ is defined by the following integral representation

$$\mathbf{A}^{1/2} = \frac{2}{\pi}\mathbf{A}\int_0^\infty (t^2\mathbf{I} + \mathbf{A})^{-1}dt \tag{2.39}$$

As for the computation itself, we use the Schur method [112, 113, 109].

2.4.4 ROM Interpolation on the Tangent space

Now that we have reviewed the preliminaries of differential geometry and matrix functions, we illustrate the interpolation of the ROMs in the context of this thesis. The matrix $\tilde{\mathbf{B}}$ in Equation 4.7 is known to be symmetric positive definite. This is because, $\tilde{\mathbf{B}} = \Phi^T\mathbf{A}^T\mathbf{A}\Phi$ and the co-variance matrix $\mathbf{A}^T\mathbf{A}$ is symmetric positive semi-definite (see [25], sec. 5.3). Additionally, multiplication by orthogonal matrix Φ of rank k where $k < rank(A)$ ensures $\tilde{\mathbf{B}}$ is symmetric positive definite. Therefore, the we are seeking a tangent plane to the manifold containing the set of all symmetric positive definite matrices of a specific size to perform our matrix interpolation.

Symmetric positive definite matrices of size $n \times n$ form a special group called the $SPD(n)$ [114, 107]. Also, for the set of all SPD matrices $\mathbf{B}_n \in \mathcal{M}$, the tangent plane is the set of all *symmetric* matrices, $\mathbf{B}'$ [107]. The geodesic connecting two points $\mathbf{B}_1$ and $\mathbf{B}_2$ for the $SPD(n)$ group is defined by the equation

$$\gamma(t) = \mathbf{B}_1^{1/2}\left(\mathbf{B}_1^{-1/2}\mathbf{B}_2\mathbf{B}_1^{-1/2}\right)^t\mathbf{B}_1^{1/2} \tag{2.40}$$

Any metric defined on the $SPD(n)$ for any two matrices uses the following functional

relationship

$$\mathcal{M}_f(\mathbf{B}_1, \mathbf{B}_2) = \mathbf{B}_1^{1/2} f\left(\mathbf{B}_1^{-1/2}\mathbf{B}_2\mathbf{B}_1^{-1/2}\right)\mathbf{B}_1^{1/2} \tag{2.41}$$

which leads to the following results for the exponential and logarithmic mapping for $SPD(n)$ [114]. In the following propositions, $\tilde{\mathbf{B}}_0$ is the anchor point and $\tilde{\mathbf{B}}'$ is the point whose mapping is desired

Proposition 2. *Exponential Mapping of* $\tilde{\mathbf{B}}'$ *from tangent plane to* $\mathcal{M}$ *at* $\tilde{\mathbf{B}}_0 \in \mathcal{M}$*, to* $\mathcal{M}$*:*

$$\mathbf{Exp}_{\tilde{\mathbf{B}}_0}\tilde{\mathbf{B}}' = \tilde{\mathbf{B}}_0^{1/2}\left(\tilde{\mathbf{B}}_0^{-1/2} exp(\tilde{\mathbf{B}}')\tilde{\mathbf{B}}_0^{-1/2}\right)\tilde{\mathbf{B}}_0^{1/2} \tag{2.42}$$

Proposition 3. *Logarithmic Mapping of* $\tilde{\mathbf{B}} \in \mathcal{M}$ *to tangent plane to* $\mathcal{M}$ *at* $\tilde{\mathbf{B}}_0 \in \mathcal{M}$*:*

$$\mathbf{Log}_{\tilde{\mathbf{B}}_0}\tilde{\mathbf{B}}' = \tilde{\mathbf{B}}_0^{1/2} log\left(\tilde{\mathbf{B}}_0^{-1/2}\tilde{\mathbf{B}}\tilde{\mathbf{B}}_0^{-1/2}\right)\tilde{\mathbf{B}}_0^{1/2} \tag{2.43}$$

The results presented in Equations 2.42 and 2.43 are used in this work to perform the tangent space interpolation.

2.4.5 Multivariate Lagrange Interpolation

Upon mapping to the tangent space, the interpolation of the ROMs are performed element-wise using polynomials in lagrange form [115, 116]. Note that only when the matrix $\tilde{\mathbf{B}}$ has parameter dependence (as in models with shape parameters) interpolation is necessary and is performed in the tangent space to manifold. In cases where there are only flow parameters (such as mach number and angle of attack), only the RHS $\tilde{\mathbf{f}}$ is required to be interpolated which is done in the Euclidean space. The procedure to interpolate with lagrange polynomials for a general multivariate case is briefly discussed as follows.

We are interested in constructing a degree n polynomial of an m-variate function $f(\mathbf{x}),\ \mathbf{x} \in \mathbb{R}^m$ which interpolates $\varrho = \binom{n+m}{m}$ points. Therefore, there needs to be atleast ϱ snapshots to interpolate the reduced matrices. We are interested in polynomials of the form $\ell_i(\mathbf{x})$ where

$$f(\mathbf{x}) = \sum_{i=1}^{\varrho} f_i \ell_i(\mathbf{x}) \tag{2.44}$$

In Equation 2.44, $\ell_i(\mathbf{x_i}) = 1$ and $\ell_i(\mathbf{x} \neq \mathbf{x_i}) = 0$ which gives $f(\mathbf{x_i}) = f_i$ and hence interpolating the true function. Before generalizing the interpolation to the multi-variate case it is easier to see the uni-variate case where $\mathbf{x_i} = x_i$. For a degree n polynomial, the (uni-variate) Lagrange interpolating polynomial is

$$\ell_i(x) = \prod_{j=1, j\neq i}^{n} \frac{x - x_j}{x_i - x_j} \tag{2.45}$$

where, when $x = x_i$, $\ell_i(x_i) = 1$ and $f(x) = f_i$; when $x \neq x_i$, $\ell_i(x_j) = 0$ at $x = x_j$. Notice that Equation 2.45 includes terms of all integral powers of x from $0, ..., n$. To generalize this aspect to the multi-variate case, we make use of the convenient idea (more so from a computational implementation point of view) of *ordered partitions* (see [117] for an introduction). For an integer i, an ordered partition is a set of non-negative integers $\leq i$ whose sum is i. More specifically, an m dimensional ordered partition of an integer i is the set of all m tuples of non-negative integers that sum to i. For an m variate lagrange poynomial of order n, we would require all ordered partitions of all integers in $[0, n]$. i.e. vectors of dimension m whose sum is $\leq n$. These integer partitions allow us to generate the family of polynomial bases for a given m and n as will be shown below. An nth degree lagrange polynomial of m variables has exactly ϱ such ordered partitions; we denote them $\mathbf{e}_1, \ldots, \mathbf{e}_\varrho,\ \ \mathbf{e}_i \in \mathbb{Z}^m,\ \ \sum_{j=1}^{m} \mathbf{e}_{ij} \leq n$. We use this generalization to write Equation 2.44 in

coefficient form as

$$f(\mathbf{x}) = \sum_{i=1}^{i=\varrho} \beta_i \left(\prod [\mathbf{x}]^{\mathbf{e}_i} \right) \tag{2.46}$$

where $[\mathbf{x}]^{\mathbf{e}_i} \in \mathbb{R}^m$ denotes the element-wise exponential of $\mathbf{x}$ with $\mathbf{e}_i$ as in $[x_1^{\mathbf{e}_{i1}}, \ldots, x_m^{\mathbf{e}_{im}}]$. Note that in Equation 2.46, the f_i and any scalar coefficients of the $x_i's$ in Equation 2.44 are lumped in β_i. Since there are ϱ interpolating points in total, there are ϱ such equations as Equation 2.46 in ϱ unknowns. Therefore, constructing the interpolant is equivalent to determining the coefficients $\boldsymbol{\beta}$ by solving the linear system ($\boldsymbol{f} = \mathbf{X}\boldsymbol{\beta}$), where $\mathbf{X}$ takes the form

$$\mathbf{X} = \begin{bmatrix} (\prod [\mathbf{x_1}]^{\mathbf{e}_1}) & \ldots & (\prod [\mathbf{x_1}]^{\mathbf{e}_\varrho}) \\ \vdots & \ddots & \vdots \\ (\prod [\mathbf{x}_\varrho]^{\mathbf{e}_1}) & \ldots & (\prod [\mathbf{x}_\varrho]^{\mathbf{e}_\varrho}) \end{bmatrix} \tag{2.47}$$

and $\boldsymbol{f} = [f_1, \ldots, f_\varrho]^T$. The interpolation is summarized by the Algorithm 1 below and an illustration is provided in Appendix E.

Algorithm 1 Multivariate Lagrange Polynomial Interpolation

Input: $\mathbf{x}, \hat{\mathbf{x}}, \boldsymbol{f}, m, n$

Output: $\hat{\boldsymbol{f}}$

1: Compute $\varrho = \binom{m+n}{m}$

2: **if** $\varrho < \text{size}(\mathbf{x}, 1)$ **then**

3: **return**

4: **end if**

5: Choose ϱ points from $\mathbf{x}$

nearest neighbors to $\hat{\mathbf{x}}$

6: Generate ϱ ordered m partitions of integers in $[0, n]$,

$[\mathbf{e_1}, \ldots, \mathbf{e}_\varrho]^T = \texttt{genPartitions}(\texttt{n}, \texttt{m})$

7: Construct $\mathbf{X}$ matrix (Eq. 2.47)

8: Solve $\mathbf{X}\boldsymbol{\beta} = \boldsymbol{f}$

9: Interpolation at $\hat{\mathbf{x}}$

$\hat{\boldsymbol{f}} = [\prod[\hat{\mathbf{x}}]^{\mathbf{e}_1}, \ldots, \prod[\hat{\mathbf{x}}]^{\mathbf{e}_\varrho}] \cdot \boldsymbol{\beta}$

2.5 ROM Solution Method

2.5.1 Sequential Quadratic Programming

The solution of the ROM is obtained via the Sequential Quadratic Programming (SQP). When it comes to constrained optimization with continuous variables, the SQP method is very popular [118]. The general idea is that the *lagrangian* function (which combines the objective function and constraints) is locally approximated as a quadratic program using the gradient and hessian information. Similarly the constraints are locally linearized. Such a quadratic objective function with linear constraints form a *quadratic program*. This is treated as a *sub-problem* that is solved at every step to find the new point along the search direction. See [119, 120, 121, 122, 123, 124] for further details.

From the computational implementation point of view, there are several parameters

Table 2.1: SQP parameters used in ROM solution

Parameter	value
Max. Func. Evals	4E+6
Max. Iters.	500
Obj. Func. tol.	1E-6
Optimality tol	1E-6
Constraint tol.	1E-6
Step tol.	1E-4
Initial Point.	nearest neighbor[3]

to the optimization algorithm that influence the overall convergence. These include (i) initial guess (ii) tolerance (iii) function evaluation limits and (iv) iteration limits. These are summarized in Table 2.1 which was fixed for all computations. Finally, the objective function and constraints are normalized for better conditioning of the problem.

In Table 2.1, Step and Objective Function tolerance refers to the relative change in the search distance and the objective function between successive iterations respectively. The Constraint tolerance refers to the upper bound of the magnitude of any constraint function at the search point. The Optimality tolerance refers to the infinity norm (maximum value) of the lagrangian at each step. The parameters are determined on several trials with the method for different problems and choosing those values that gave best results.

2.6 Computational Cost

The overall computational cost is dominated by the offline phase where the model is built. In this section, we aim to provide an estimate of the computational cost in terms of Floating Point OPerationS (FLOPS) necessary to build the ROM, given the snapshots (training data from high-fidelity simulations). The cost of the online phase is trivial comparatively and the wall-clock time is more relevant in this scenario. The off-line phase includes the following steps and we discuss the

1. Snapshot scaling

2. Proper Orthogonal Decomposition

3. Finite Volume Discreitization

4. Projection

2.6.1 Snapshot scaling

The snapshots of each observable might have a different scale; this is because the primitives such as pressure is typically in $\mathcal{O}(10^5 - 10^6)$ while x-velocity is in $\mathcal{O}(10^2 - 10^3)$ and hence the observables derived from primitives will inherit this. Additionally, the range of given variable can also be large in flows with large gradients (such as transonic flows). Therefore for a more uniform performance of the SVD while extracting the POD basis, it is important to scale the data such that they have similar ranges. In this work, we scale the data such that all snapshots have a range of $[0, 1]$. Each snapshot is scaled as

$$\mathbf{y_i}_{scaled} = \frac{\mathbf{y_i} - \texttt{min}([\mathbf{y}_1, ..., \mathbf{y}_M])}{\texttt{max}([\mathbf{y}_1, ..., \mathbf{y}_M]) - \texttt{min}([\mathbf{y}_1, ..., \mathbf{y}_M])} \tag{2.48}$$

where the `max` and `min` operators return a scalar that is the maximum and minimum across all snapshots. This is a $2N$ operation that is repeated for every observable; so a total of $2NO$ operations where O is the problem dependent number of observables.

2.6.2 POD basis extraction

In the present approach, the POD bases are individually extracted for each observable snapshot set and assembled into a block-diagonal matrix, $\mathbf{\Phi}$. Since thin SVD is performed on each of the snapshot set, the cost per $(N \times M)$ matrix is $\mathcal{O}(N^2 M)$ [25] and hence a total of $\mathcal{O}\left(N^2 MO\right)$. [4]

[4] It should be noted that from a wall-clock time perspective, the POD step is not the most expensive due to efficient implementations of the SVD available in numerical libraries such as Matlab or SciPy.

2.6.3 Finite Volume Discretization

The discretization of the linear differential operator is the dominating step in the offline phase of the present methodology. For problems involving shape parameters, a unique linear operator is constructed for each snapshot and the projected matrix (reduced order matrix) has to be stored. The complexity of approximating the full-order linear operator scales as $\mathcal{O}(N)$ for both the diffusion and gradient operators; see [152] for details. Further, since this is repeated for each snapshot, the total cost scales as $\mathcal{O}(NM)$.

2.6.4 Projection

The projection step involves matrix multiplication involving the full-order linear operator and the trial basis matrix. We are specifically interested in the product

$$\tilde{\mathbf{B}} = \mathbf{\Phi}^T \mathbf{A}^T \mathbf{A} \mathbf{\Phi}$$

$\mathbf{A}$ is of size $SN \times ON$ [5] and has a highly sparse structure [49]. Therefore, the product $\mathbf{A}^T\mathbf{A}$ has $\mathcal{O}(OSN^2)$ operations (note that a dense matrix multiplication of equivalent size would take approximately $\mathcal{O}(ON^3)$). The product $\mathbf{\Phi}^T\mathbf{A}^T\mathbf{A}$ takes $\mathcal{O}((ON)^2k)$ operations while the product $[\mathbf{\Phi}^T\mathbf{A}^T\mathbf{A}]\mathbf{\Phi}$ takes $\mathcal{O}(ONk^2)$ operations. Therefore in total (for all snapshots), the projection step involves $\mathcal{O}\left(M \times (OSN^2 + (ON)^2)k + ONk^2)\right)$ operations.

Similarly the projection of the RHS

$$\tilde{\mathbf{f}} = \mathbf{\Phi}^T \mathbf{A}^T \mathbf{f}$$

involves $\mathcal{O}(SNk)$ operations per snapshot and hence a total of $\mathcal{O}(MSNk)$.

[5] recall that O is the total number of observables and S is the number of PDEs in the coupled system that represents the FOM

Table 2.2: Summary of offline computational cost

Operation	FLOPS
Snapshot Scaling	$\mathcal{O}(N)$
POD	$\mathcal{O}(N^2M)$
Finite Vol. Discret.	$\mathcal{O}(MN)$
Projection	$\mathcal{O}(N^2M)$

2.6.5 Summary of Offline cost

The summary of computational cost are provided in Table 2.2. Note that only the dominating factors of the cost are provided in the table. It can be seen that the most expensive steps of the method are the POD and the projection which scale as $\sim N^2M$; as N increases the cost of these steps increases quite rapidly. The finite volume discretization is relatively a cheaper step that scales linearly with grid size. Additionally, this step is performed only once (cost $\mathcal{O}(N)$) when linear operator has affine parameter dependence, as in problems with flow parameters.

2.7 Overall Framework & Algorithm

The overall method is summarized in Algorithm 2 and in Figure 2.7. Note that in the algorithm, D represents the design space while D_{train} represents a subset of D used for snapshot generation (model training).

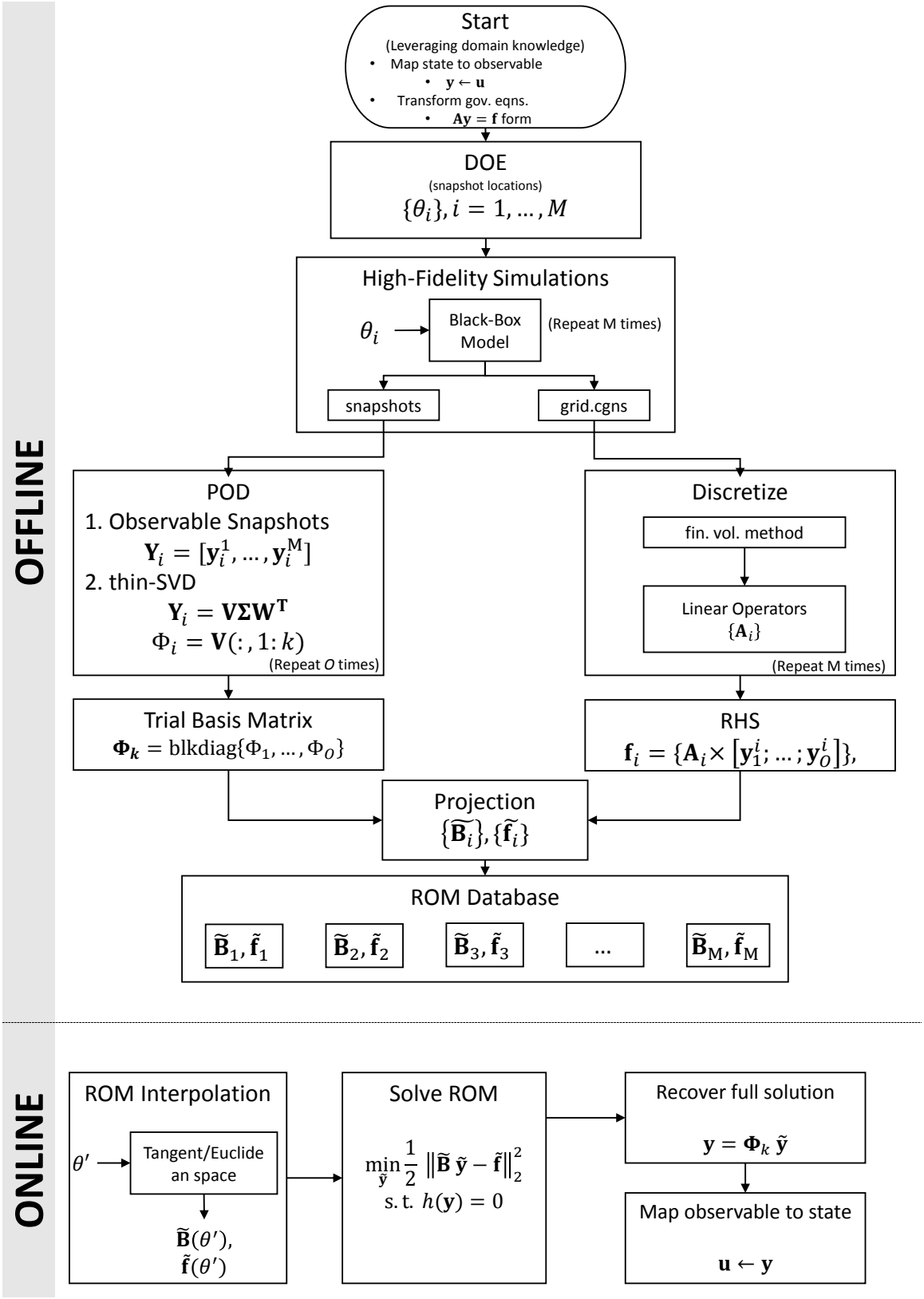


Figure 2.7: Schematic representation of the overall framework

Algorithm 2 Non-Intrusive Projection-Based Model Reduction

0: Choose M snapshot locations for model training $\boldsymbol{\theta}_i \in D_{train} \subset D$

% **OFFLINE**

% Solve FOM, construct snapshot matrix and extract POD bases:

1: **for** $i = 1$ to O **do**

2: $\mathbf{Y}_i = [\mathbf{y}_i^1, \ldots, \mathbf{y}_i^M] \in \mathbb{R}^{N \times M}$

3: $\mathbf{Y}_i = \mathbf{V}\Sigma\mathbf{W}^T, \ \ (\texttt{thin} - \texttt{SVD})$

4: $\Phi_i = \mathbf{V}(:, 1 : k_i) \in \mathbb{R}^{N \times k_i}$

5: **end for**

6: $\boldsymbol{\Phi}_k = \texttt{blkdiag}\{\Phi_1, \ldots, \Phi_O\} \in \mathbb{R}^{N \times k}$ % trial basis matrix

% Construct system matrices

7: **for** $\boldsymbol{\theta_i}$, $i = 1$ to M **do**

8: $\mathbf{A} \leftarrow$ Discretize Linear Operator

9: $\mathbf{f} \leftarrow \mathbf{A} \times \mathbf{y}$

10: $\tilde{\mathbf{B}} \leftarrow \Phi_i^T(\mathbf{A}^T\mathbf{A})\Phi_i$

11: $\tilde{\mathbf{f}} \leftarrow \Phi_i^T\mathbf{A}^T\mathbf{f}$

12: **end for**

% **ONLINE**

% Prediction: for any $\boldsymbol{\theta}' \notin D_{train}, \ \boldsymbol{\theta}' \in D$

13: interpolate element-wise $\tilde{\mathbf{B}}(\boldsymbol{\theta}')$, $\tilde{\mathbf{f}}(\boldsymbol{\theta}')$

Algorithm 1

14: Solve ROM:

$$\underset{\tilde{\mathbf{y}}}{\text{minimize}} \quad \frac{1}{2}\|\tilde{\mathbf{B}}\tilde{\mathbf{y}} - \tilde{\mathbf{f}}\|_2^2$$

$$\text{s.t.} \qquad h(\mathbf{y}) = 0$$

15: Project ROM onto FOM space: $\mathbf{y} = \Phi_k\tilde{\mathbf{y}}$

16: Map observables back to state variables

$\mathbf{u} \leftarrow \mathbf{y}$

CHAPTER 3

APPLICATION: CANONICAL PDE

As a first step towards demonstrating the proposed ROM approach, canonical parametric PDEs are used. Such PDEs are a simplified representative of the non-linear, static, parametric systems that is ultimately of interest in this thesis. Additionally, these PDEs offer the convenience of testing the method on problems with a relatively small grid size. Two specific types of PDEs are used in this chapter one that is linear and one that is non-linear with an exponential type non-linearity. In this chapter, a formal illustration of all the steps described in Chapter 2 is provided.

Research Question 1. *Is the proposed approach feasible?*

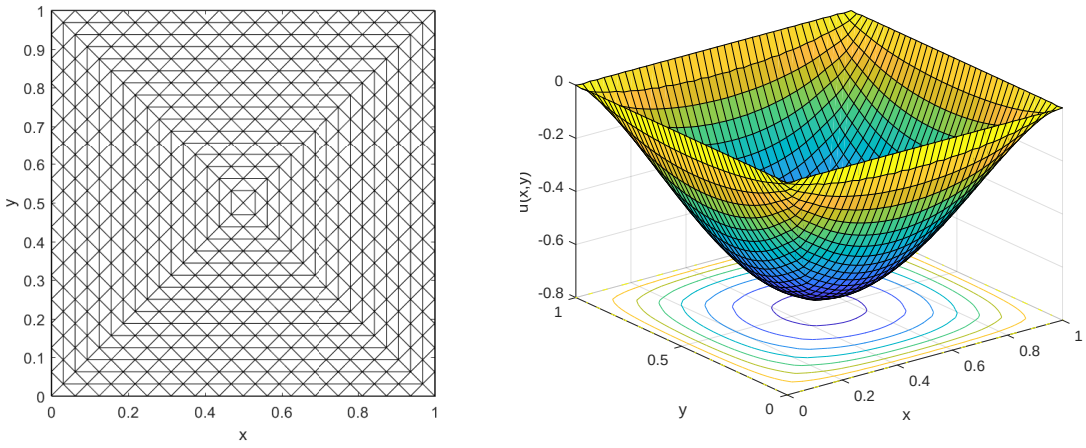
3.1 Linear Parametric PDE

The linear test case is the Poisson's equation with Dirichlet boundary conditions given below

$$\begin{aligned} -\nabla \cdot (\Gamma \nabla u(x,y)) &= \theta \\ u &= 0 \text{ at boundary} \end{aligned} \tag{3.1}$$

where, the spatial variables $(x, y) \in (0, 1)^2$ and the parameter $\theta \in [1.0,\ 10.0]$ and the diffusion coefficient, $\Gamma = 1$. The Matlab PDE Toolbox [125] is the black-box code used in this test case to obtain the snapshots. The domain is discretized with an unstructured mesh comprising 1024 triangular cells ($N = 1024$). The computational grid and a sample snapshot solution of the PDE is shown in Figure 3.1.

A total of 20 snapshots locations were generated using using a latin hyper cube sam-



(a) Computational domain and mesh. 1024 triangular cells, 545 nodes and 1568 faces

(b) Solution of linear test case at $\theta = 10$

Figure 3.1: Computational domain and sample solution for the test cases

pling.

The ROM is constructed and solved via the following steps

1. Step-1: Linearize

   Original PDE: $\nabla^2 u = \theta$

   Map state to observable: $y \leftarrow u$

   Linearized PDE: $\nabla^2 y = \theta$ (unchanged for linear systems)

   Solve original (high-fidelity) PDE system and obtain snapshots of y.

   Extract POD basis, $\mathbf{\Phi}_k$

2. Step-2: Discretize

   Inputs: computational grid, Linear term (∇^2)

   Output: Linear operator $\mathbf{A}$

   Extract RHS: $\mathbf{f} = \mathbf{A}\mathbf{y}$

3. Step-3: Projection

   $\mathbf{y} = \mathbf{\Phi}_k \tilde{\mathbf{y}}$

   $\mathbf{\Phi}_k^T \mathbf{A}^T \mathbf{A} \mathbf{\Phi}_k \tilde{\mathbf{y}} = \mathbf{\Phi}_k^T \mathbf{A}^T \mathbf{f}$

   $\tilde{\mathbf{B}} \tilde{\mathbf{y}} = \tilde{\mathbf{f}}$

   Repeat for every parameter snapshot, resulting in ROM database

4. Step-4: ROM interpolation

   Inputs: ROM database, new parameter instance

   Output: Interpolated ROM $\tilde{\mathbf{B}}, \tilde{\mathbf{f}}$

5. Step-5: Solve ROM

   $\underset{\tilde{\mathbf{y}}}{\text{minimize}} \frac{1}{2} \|\tilde{\mathbf{B}} \tilde{\mathbf{y}} - \tilde{\mathbf{f}}\|_2^2$

Due to the lack of any non-linear terms, the FOM in Equation 3.1 is trivially transformed to the observable as $y \leftarrow u$. This leads to the transformed equation

$$-\nabla^2 y = \theta \tag{3.2}$$

which upon discretization leads to

$$-\mathbf{L}\mathbf{y} + \mathbf{b}_a = \theta \times \mathbf{1} \tag{3.3}$$

where $\mathbf{L} \in \mathbb{R}^{N \times N}$ is the discrete laplacian and $\mathbf{1}$ is a vector of ones of size N.

$$\underbrace{-\mathbf{L}}_{\mathbf{A}} \mathbf{y} = \underbrace{-\mathbf{b}_a + \theta \times \mathbf{1}}_{\mathbf{f}} \tag{3.4}$$

Model reduction is performed on the system in Equation 3.4 as explained in Chapter 2.

Table 3.1: Relative error of the ROM validation points for the linear canonical PDE test case

Validation Case	θ	R.E.	Validation Case	θ	R.E.
1	3.79	13.29E-12%	7	5.50	13.93E-12%
2	8.12	12.56E-12%	8	6.22	11.88E-12%
3	5.33	12.56E-12%	9	5.87	16.38E-12%
4	3.51	14.40E-12%	10	2.08	17.49E-12%
5	9.39	12.41E-12%	11	3.01	13.91E-12%
6	8.76	12.41E-12%	12	4.71	16.14E-12%

The ROM is of size 20×20 which is significantly smaller than the original system that is 1024×1024. The RHS of the FOM is linearly interpolated for parametric changes while the LHS (linear operator) is precomputed for this test case. The ROM is solved via a direct method to solve linear systems.

Although trivial, this test case was necessary as a simple first step to evaluate the methodology. The overlaid contour plots of the ROM and FOM for the validation points are shown in Figure 3.2 and the relative errors are summarized in Table 3.1. The relative error was $\in \mathcal{O}(1E-12)\%$ consistently indicating very good agreement. The fact that the parametric dependence is linear and that there are no non-linear terms in the FOM, leads to very high accuracy in the ROM. The following test case however, pushes this envelope to include non-linear dependence of both the state and parameters.

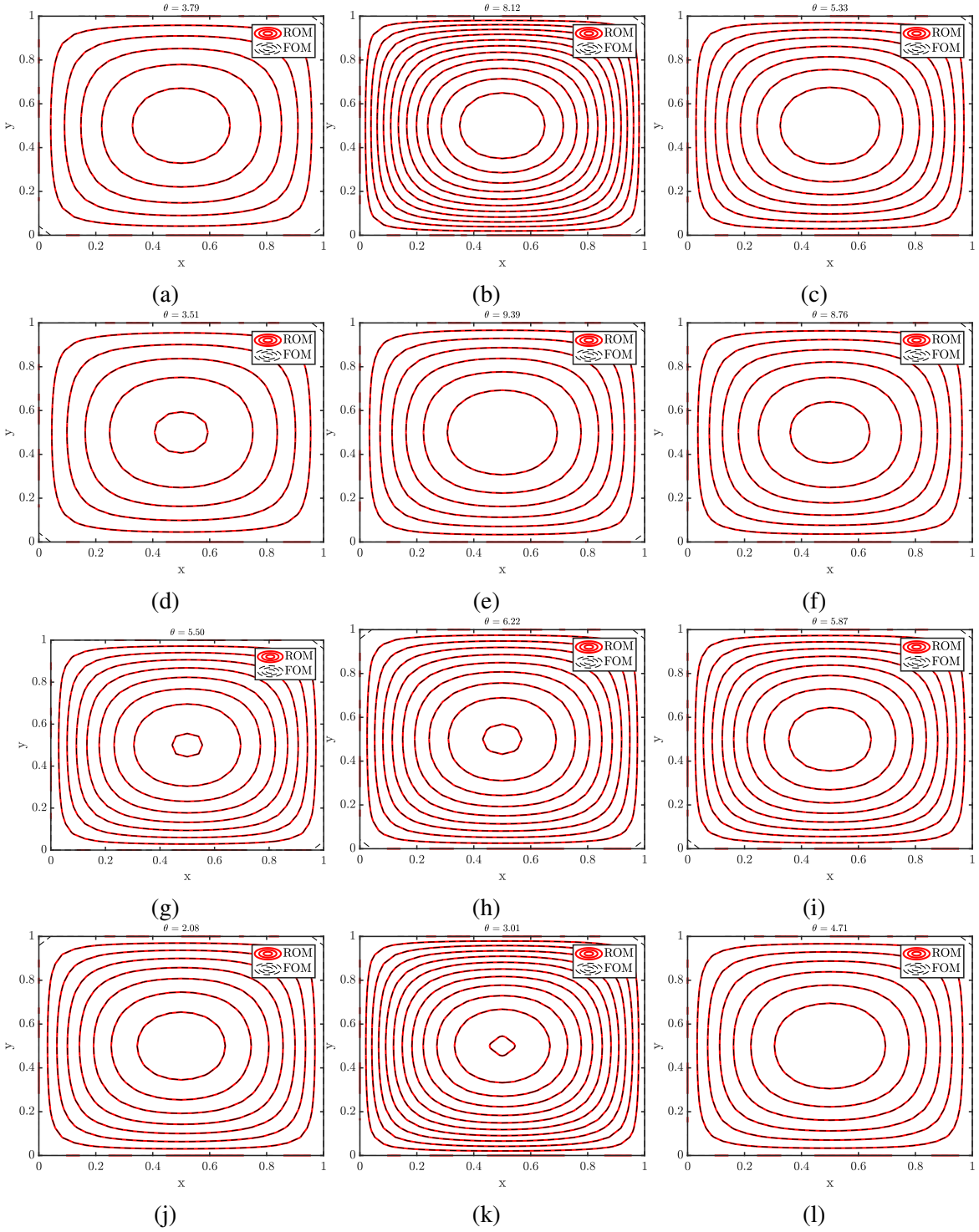


Figure 3.2: Validation of ROM against FOM for the linear PDE test case.

3.2 Non-Linear Parametric PDE

The canonical PDE for the non-linear test case is the same as that used in [126, 47] and is given below

$$\begin{aligned} -\nabla^2 u(x,y) + s(u(x,y);\boldsymbol{\theta}) &= 100 sin(2\pi x) sin(2\pi y) \\ s(u;\boldsymbol{\theta}) &= \frac{\theta_1}{\theta_2}(e^{u\theta_2} - 1) \\ \text{BC: } u &= 0 \text{ at boundary} \end{aligned} \tag{3.5}$$

where, the spatial variables $(x, y) \in (0, 1)^2$ and the parameter $\boldsymbol{\theta} = [\theta_1, \theta_2] \in [0.01,\ 2.0]^2$. The PDE is solved with Dirichlet boundary conditions of $u = 0$ along the boundaries. The Matlab PDE Toolbox [125] is the black-box code used in this test case to obtain the snapshots. The domain is discretized with an unstructured mesh comprising 1024 triangular cells ($N = 1024$). The computational grid and a sample snapshot solution of the PDE is shown in Figure 3.3.

The ROM is constructed and solved via the following steps

1. Step-1: Linearize

   Original PDE: $-\nabla^2 u(x,y) + \frac{\theta_1}{\theta_2}(e^{u\theta_2} - 1) = 100 sin(2\pi x) sin(2\pi y)$

   Map state to observable: $[y_1,\ y_2] \leftarrow [u,\ \frac{\theta_1}{\theta_2}\left(e^{\theta_2 u} - 1\right)]$

   Linearized PDE: $\begin{bmatrix} -\nabla^2 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = 100 sin(2\pi x) sin(2\pi y)$

   Solve original (high-fidelity) PDE system and obtain snapshots of y_1 and y_2.

   Extract POD basis, $\boldsymbol{\Phi}_k$

2. Step-2: Discretize

Inputs: computational grid, Linear terms ($[-\nabla^2,\ 1]$)

Output: Linear operator $\mathbf{A}$

Extract RHS: $\mathbf{f} = \mathbf{A}\mathbf{y}$

3. Step-3: Projection

$\mathbf{y} = \mathbf{\Phi}_k \tilde{\mathbf{y}}$

$\mathbf{\Phi}_k^T \mathbf{A}^T \mathbf{A} \mathbf{\Phi}_k \tilde{\mathbf{y}} = \mathbf{\Phi}_k^T \mathbf{A}^T \mathbf{f}$

$\tilde{\mathbf{B}} \tilde{\mathbf{y}} = \tilde{\mathbf{f}}$

Repeat for every parameter snapshot, resulting in ROM database

4. Step-4: ROM interpolation

Inputs: ROM database, new parameter instance

Output: Interpolated ROM $\tilde{\mathbf{B}}, \tilde{\mathbf{f}}$

5. Step-5: Solve ROM

$\underset{\tilde{\mathbf{y}}}{\text{minimize}} \frac{1}{2} \|\tilde{\mathbf{B}} \tilde{\mathbf{y}} - \tilde{\mathbf{f}}\|_2^2$

s.t. constraint $h_1(\mathbf{y}) = \mathbf{y}_2 - \frac{\theta_1}{\theta_2} \left(e^{\theta_2 \mathbf{y}_1} - 1 \right) = 0$ (to close under-determined system - 1 PDE, 2 unknowns y_1, y_2)

The following transformation is done to the PDE $[y_1,\ y_2] \leftarrow [u,\ \frac{\theta_1}{\theta_2} \left(e^{\theta_2 u} - 1 \right)]$ which leads to the modified equation in terms of the observables

$$\begin{bmatrix} -\nabla^2 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = 100 sin(2\pi x) sin(2\pi y) \tag{3.6}$$

where, ∇^2 represents the laplacian. Upon discretization, the above equation transforms

to

$$\begin{bmatrix} -\mathbf{L} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix} + \mathbf{b}_a = 100sin(2\pi\mathbf{x})sin(2\pi\mathbf{y}) \tag{3.7}$$

where $\mathbf{L} \in \mathbb{R}^{N \times N}$ is the discrete laplacian and $\mathbf{I}$ is the identity matrix.

$$\underbrace{\begin{bmatrix} -\mathbf{L} & \mathbf{I} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}}_{\mathbf{y}} = \underbrace{-\mathbf{b}_a + 100sin(2\pi\mathbf{x})sin(2\pi\mathbf{y})}_{\mathbf{f}} \tag{3.8}$$

which reduces to the $\mathbf{Ay} = \mathbf{f}$ form that we are interested in. The ROM is developed from this point using the same approach discussed in Chapter 2.

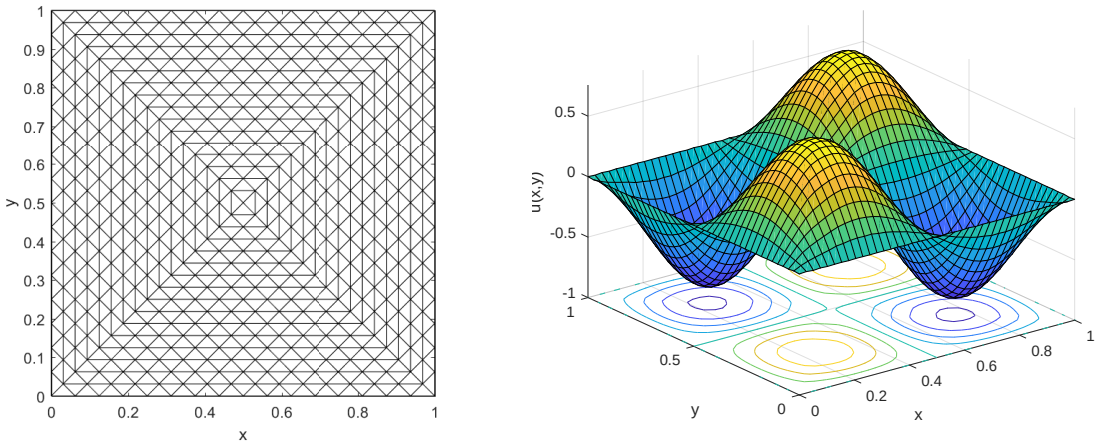
In this test case, the observable y_2 is chosen to be $\frac{\theta_1}{\theta_2}(e^{\theta_2 u} - 1)$ and thereby lumping the parameter dependence along with the observable. This makes the linear operator independent of parameters and hence can be pre-computed. However, with a different choice of y_2 such as $e^{\theta_2 u}$, the linear operator, $\mathbf{A}$ becomes dependent on parameters, and therefore cannot be pre-computed. In such a case, a unique matrix is constructed for each snapshot, similar to the RHS, $\mathbf{f}$, and the linear operator for new parameter instances can be interpolated by mapping to the tangent space, as explained in Chapter 2 (section 2.4.4) element-wise using methods in [127] or [128].

Since there is one observable in excess of the number of PDEs in the system (i.e. $S = 1,\ O = 2$, see Chapter 2), the only constraint for the ROM for this case is given as follows

$$h_1(\mathbf{y}) = \mathbf{y}_2 - \frac{\theta_1}{\theta_2}\left(e^{\theta_2 \mathbf{y}_1} - 1\right) = 0 \tag{3.9}$$

The non-linear constraint equation above enforces the relationship between the two observables which should hold true. and is efficiently computed using the DEIM described in Appendix B.

In order to extract the POD basis, 20 snapshot locations were computed that vary the



(a) Computational domain and mesh. 1024 triangular cells, 545 nodes and 1568 faces

(b) Solution of non-linear test case at $\boldsymbol{\theta} = [0.3, 0.9]$

Figure 3.3: Computational domain and sample solution for the nonlinear PDE test case

parameters (θ_1, θ_2) using a Latin Hypercube Design [4]. Then the black-box code is run to generate the snapshots of the 2 observables, $\mathbf{y_1}$ and $\mathbf{y_2}$. The singular values of the resulting snapshot matrix were not truncated to retain maximum accuracy. This results in a reduced system that is 40×40 which is still significantly smaller than the original system which is 1024×1024.

The comparison of the ROM and FOM solutions for a parameters outside of those used in the snapshots is shown in Figure 3.4. The linear operator matrix was pre-computed since it is independent of parameters while the RHS vector was piece-wise interpolated with a bi-variate 3^{rd} order lagrange polynomial. The relative error (R.E.) is computed as the relative value of the 2-norm error between the full and reduced order model solutions (Equation 3.10).

$$R.E. = \frac{\|FOM - ROM\|_2}{\|FOM\|_2} \tag{3.10}$$

The ROM results agree with an average relative error of $\approx 1.2\%$ (see Table 3.2) which is an error measure based on the prediction through the entire computational domain, ver-

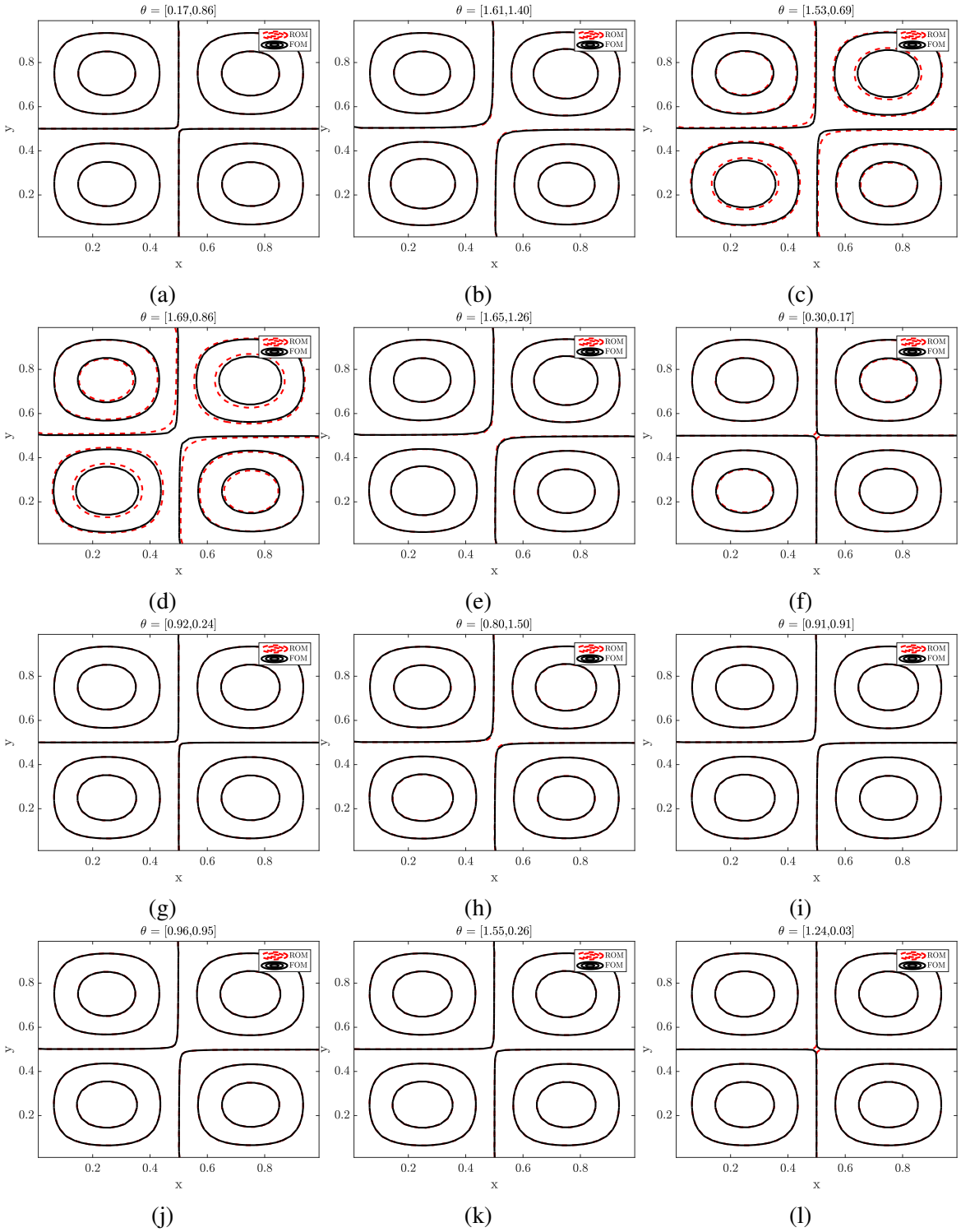


Figure 3.4: Comparison of ROM (solid lines) and FOM (dashed lines) for the canonical PDE validation cases.

Table 3.2: Relative error of the ROM validation points for the Canonical PDE test case

Validation Case	θ_1	θ_2	R.E.	Validation Case	θ_1	θ_2	R.E.
1	0.17	0.86	0.08 %	7	0.92	0.24	0.04 %
2	1.61	1.40	0.44 %	8	0.80	1.50	0.55 %
3	1.53	0.69	4.39 %	9	0.91	0.91	0.08 %
4	1.69	0.86	6.74 %	10	0.96	0.95	0.20 %
5	1.65	1.26	0.51 %	11	1.55	0.26	0.06 %
6	0.30	0.17	0.95 %	12	1.24	0.03	0.19 %

ifying that the proposed methodology is capable of accomplishing high levels of accuracy. The two main sources of error in this test case are (i) the approximation of the linear operator and (ii) the DEIM interpolation of the RHS vector. Since the Matlab PDE toolbox used to obtain the snapshots is a black-box it is not possible to quantify the error due to the linear operator approximation, and hence the overall error of approximation. However, given that the $R.E.$ for this test case was consistently in the $\mathcal{O}(1)\%$ which is a typically accepted range of accuracy of surrogate models used for engineering design optimization, it is concluded that the present method does approximately satisfy the governing equations at the ROM level, thereby serving as a physics-based surrogate model that is computationally cheap. As a next step, the method is applied towards approximating the flow past an airfoil, governed by the compressible Euler equations.

3.3 Discussion

The canonical test cases allowed to demonstrate the methodology and prove that it verifies that with the proposed methodology, projection-based model reduction of non-linear parametric system is feasible and the accuracy of the results and the computational efficiency of the model, show its suitability towards applying them to applications in the many-query context. Overall, the non-linear test case incurred slightly higher error than the linear case which is expected. However, while most practical PDE system involves a polynomial-type non-linearity (such as the Navier-Stokes equations), the test case chosen here had an exponential non-linearity.

In the following chapter, we demonstrated the method on a more advanced PDE system - the compressible Euler equations. The key difference compared to the canonical test cases is that (i) we consider a coupled system non-linear PDEs, (ii) the grid sizes are significantly larger, (iii) the parameter dimensionality is extended upto 8 parameters and, (iv) the test cases include flow regimes that could lead to discontinuities such as shocks. Therefore, we demonstrate the method on problems that are more representative of practical applications. Finally, in the canonical test cases, the boundary conditions were explicitly handled since it was easier to do so. In the following chapter, boundary conditions are lumped into the RHS of the ROM in order to keep the approach relatively non-intrusive as we scale it up to larger and more practical problems.

CHAPTER 4

APPLICATION: COMPRESSIBLE EULER EQUATIONS

The method is now implemented on the Euler equations governing the 2D, compressible, inviscid flow past an airfoil. Two types of parameters are considered namely (i) *flow parameters*: the Mach number ($\mathbb{M}$) and the angle of attack (α) and (ii) *geometry parameters*: the airfoil shape parameters. Each parameterization is considered separately in this chapter and the flow snapshots are generated by solving the coupled PDE system in the commercial black-box CFD solver, STARCCM+ [129]. The computational grid used by the CFD solver is exported in the CGNS [130] format and used to approximate the linear operator matrix in the present method via the finite volume method. We begin by presenting the 2D compressible version of the Euler equations in conservation form

$$\nabla_x \mathbf{F} + \nabla_y \mathbf{G} = 0 \tag{4.1}$$

where

$$\mathbf{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uH \end{bmatrix}, \ \mathbf{G} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vH \end{bmatrix}$$

$$H = E + \frac{p}{\rho}$$

$$\rho E = \frac{1}{2}\rho(u^2 + v^2) + \frac{p}{\gamma - 1}$$

and ∇_x and ∇_y are the x and y components of the gradient ∇ respectively. The following transformation is then performed

$$[\rho u, \rho v, \rho uv, p, \rho u^2, \rho v^2, \rho uH, \rho vH]^T \rightarrow [y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8]^T$$

from the state variables to observables, leading to the transformed equation below.

$$\begin{bmatrix} \nabla_x & \nabla_y & & & & & & \\ & & \nabla_y & \nabla_x & \nabla_x & & & \\ & & \nabla_x & \nabla_y & & \nabla_y & & \\ & & & & & & \nabla_x & \nabla_y \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \end{bmatrix} = \mathbf{0} \tag{4.2}$$

where in the above equation, empty spaces in the matrix denote zeros. The equation upon discretization leads to

$$\underbrace{\begin{bmatrix} \mathbf{G}_x & \mathbf{G}_y & & & & & & \\ & & \mathbf{G}_y & \mathbf{G}_x & \mathbf{G}_x & & & \\ & & \mathbf{G}_x & \mathbf{G}_y & & \mathbf{G}_y & & \\ & & & & & & \mathbf{G}_x & \mathbf{G}_y \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_3 \\ \mathbf{y}_4 \\ \mathbf{y}_5 \\ \mathbf{y}_6 \\ \mathbf{y}_7 \\ \mathbf{y}_8 \end{bmatrix}}_{\mathbf{y}} = - \underbrace{\begin{bmatrix} \mathbf{b}_{a1} \\ \mathbf{b}_{a2} \\ \mathbf{b}_{a3} \\ \mathbf{b}_{a4} \\ \mathbf{b}_{a5} \\ \mathbf{b}_{a6} \\ \mathbf{b}_{a7} \\ \mathbf{b}_{a8} \end{bmatrix}}_{\mathbf{f}} \tag{4.3}$$

where, $\mathbf{G}_x$ and $\mathbf{G}_y$ represents the discrete version gradient operators ∇_x and ∇_y respectively and again, the empty spaces if the matrix denote block matrices of zeros. With the FOM reduced to the $\mathbf{A}\mathbf{y} = \mathbf{f}$ form and $\mathbf{A} \in \mathbb{R}^{4N \times 8N},\ \mathbf{y}, \mathbf{f} \in \mathbb{R}^{8N}$, Equation 4.3 represents and under-determined system. Therefore they are closed using non-linear constraints given by Equation 4.4. Notice that the constraints express the relationship between the first $S = 4$ observables (y_1 through y_4) and the remaining $O - S;\ (O = 8)$ observables (y_5 through

y_8). It should be noted that all the observables that are in excess of the number of equations can be expressed as some function of the rest. The constraints are expressed in terms of the continuous form of the state and observable below.

$$\begin{aligned} h_1 &= \rho u^2 - \frac{(\rho u)(\rho u v)}{\rho v} \equiv y_5 - \frac{y_1 y_3}{y_2} = 0 \\ h_2 &= \rho v^2 - \frac{(\rho v)(\rho u v)}{\rho u} \equiv y_6 - \frac{y_2 y_3}{y_1} = 0 \\ h_3 &= \rho u H - \rho u \left(E + \frac{p}{\rho} \right) \equiv y_7 - y_1 \left(E + \frac{y_4 y_3}{y_1 y_2} \right) = 0 \\ h_4 &= \rho v H - \rho v \left(E + \frac{p}{\rho} \right) \equiv y_8 - y_2 \left(E + \frac{y_4 y_3}{y_1 y_2} \right) = 0 \end{aligned} \tag{4.4}$$

where the energy term, E can be expressed in terms of $y_i's$ as follows

$$E = \frac{1}{2} \left(\frac{y_3 y_5}{y_1 y_2} + \frac{y_3 y_6}{y_1 y_2} \right) + \frac{y_3 y_4}{y_1 y_2 (\gamma - 1)}$$

With the transformed governing equations (in terms of observables), we proceed with the Petrov-Galerkin projection step with trial basis $\mathbf{\Phi} \in \mathbb{R}^{8N \times k}$ and test basis $\mathbf{\Psi} = \mathbf{A}\mathbf{\Phi}$,

$$\mathbf{\Phi}^T \mathbf{A}^T \mathbf{A} \mathbf{\Phi} \tilde{\mathbf{y}} = \mathbf{\Phi}^T \mathbf{A}^T \mathbf{f} \tag{4.5}$$

where

$$\mathbf{\Phi} = \begin{bmatrix} \Phi_1 & & \\ & \ddots & \\ & & \Phi_8 \end{bmatrix} \in \mathbb{R}^{8N \times k}$$

and $k = \sum_i k_i,\ i = 1, .., 8$ and each k_i represents the number of POD modes of the respective observables $\mathbf{y}_i$ required to capture certain specified amount of energy. The projection step above leads to the reduced system

$$\tilde{\mathbf{B}} \tilde{\mathbf{y}} = \tilde{\mathbf{f}} \tag{4.6}$$

The final ROM is the solution of the following minimization problem given as

$$\begin{aligned} \underset{\tilde{\mathbf{y}}}{\text{minimize}} \quad & \frac{1}{2}\|\tilde{\mathbf{B}}\tilde{\mathbf{y}} - \tilde{\mathbf{f}}\|_2^2 \\ \text{s.t.} \quad & h_i(\mathbf{y}) = 0,\ i = 1, \ldots, 4 \end{aligned} \tag{4.7}$$

where the $h_i(\mathbf{y})\,'s$ are given in Equation 4.4. The constraints by themselves pose a serious limitation to the computational efficiency of the method, but can be handled well by the DEIM. The overview of the DEIM along with expansion of the constraints using Equation 4.4 is provided in Appendix B. The Sequential Least Squares Quadratic Programming (SLSQP) [131] is used to solve the resulting non-linear constrained optimization problem and the results are discussed as follows.

For the results presented in this chapter, the following error metrics are used

$$\begin{aligned} C_P \text{ Error} &= \frac{\left\|C_P^{FOM} - C_P^{ROM}\right\|_\infty}{\left\|C_P^{FOM}\right\|_\infty} \times 100 \\ C_d \text{ Error} &= \frac{\left|C_d^{FOM} - C_d^{ROM}\right|}{C_d^{FOM}} \times 100 \\ C_l \text{ Error} &= \frac{\left|C_l^{FOM} - C_l^{ROM}\right|}{C_l^{FOM}} \times 100 \end{aligned} \tag{4.8}$$

4.1 Variation in Flow Parameters

As mentioned earlier, as a first step the flow parameters $(\mathbb{M}, \alpha)$ are varied keeping the geometry fixed. This section presents the comparison of the prediction of the full state via the ROM on the validation points against FOM solution.

4.1.1 NACA0012

Research Question 2. *Can the proposed approach predict the field variables accurately?*

The NACA0012 is a symmetric airfoil given by Equation 4.9, shown in Figure 4.2 and is the first test case used in this work. The parameter ranges are set as $\mathbb{M} \in [0.3, 0.6]$ and $\alpha \in [0, 3]\ deg$ and 45 snapshots were generated using a maximin latin hypercube design out of which 40 were used to build the model while 5 were used to validate the model, see Figure 4.1. The freestream conditions for this test case are summarized in Table 4.1.

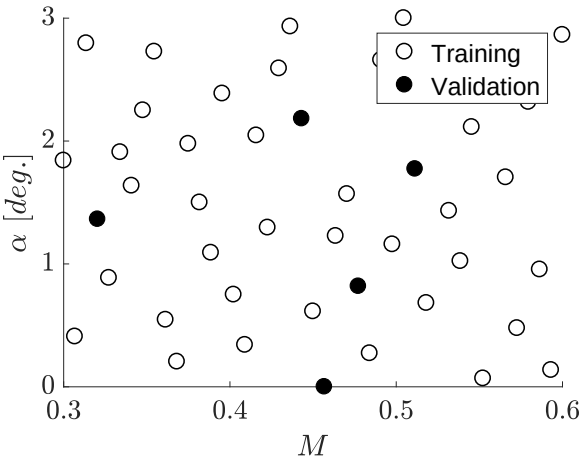


Figure 4.1: Parameter snapshots for the NACA0012 test case

$$y = \pm\frac{0.12}{0.2}\left(0.2969\sqrt{x} - 0.1260x - 0.3516x^2 + 0.2843x^3 - 0.1015x^4\right) \tag{4.9}$$

An unstructured mesh was generated to discretize a circular domain that is 150 chord lengths around it with a total of 11,265 triangular cells, 5776 vertices, shown in Figures 4.3 and 4.4. The singular values that capture upto 99.99% of the variation of the observables were retained leading to a reduced order system of size $k \times k,\ k = 148$. Note that the original system is of size $8N \times 8N$ where $N = 11,265$ for the present test case and hence the dimensionality reduction is significant.

The relative error is calculated for the 5 validation points in Figure 4.1 and are summarized in Table 4.2. Note that these errors are calculated based on all 8 observables concatenated as a single vector. Overall, it is observed that the relative error of prediction is $\mathcal{O}(1)\ \%$ similar to the canonical PDE test case.

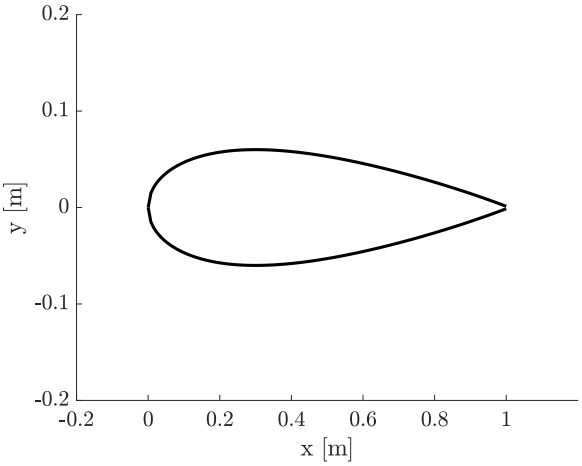


Figure 4.2: The NACA0012 airfoil shape

Table 4.1: Free-stream conditions for the NACA0012 test case

P_∞	101,325 Pa
ρ_∞	1.225 kg/m^3
a_∞	340.296 m/s
μ_∞	1.785E-5 $Pa-s$

The pressure and mach number contours for the validation cases 1 & 5 are shown in Figures 4.5 and 4.6. The ability of the ROM to capture the spatial distribution of the field variables is emphasized. A more rigorous comparison can be made by overlaying contour plots of the ROM and FOM solution, which is done in Figure 4.7 in terms of the pressure and mach number contours. The mach number itself is calculated as $\mathbb{M} = \sqrt{\frac{u^2+v^2}{\gamma\ p/\rho}}$ and hence comparing p and $\mathbb{M}$ contours, albeit in an indirect way includes the comparison of all 4 primitive variables namely p, ρ, u, v.

The plots show that the ROM prediction compares very well with the FOM results in

Table 4.2: Relative error of the ROM validation points for the NACA0012 test case

Validation Case	$\mathbb{M}$	$\alpha\ [deg.]$	Rel. Error
1	0.51	1.77	1.008 %
2	0.477	0.82	0.128 %
3	0.32	1.36	0.69 %
4	0.44	2.18	0.15 %
5	0.46	0	0.194 %

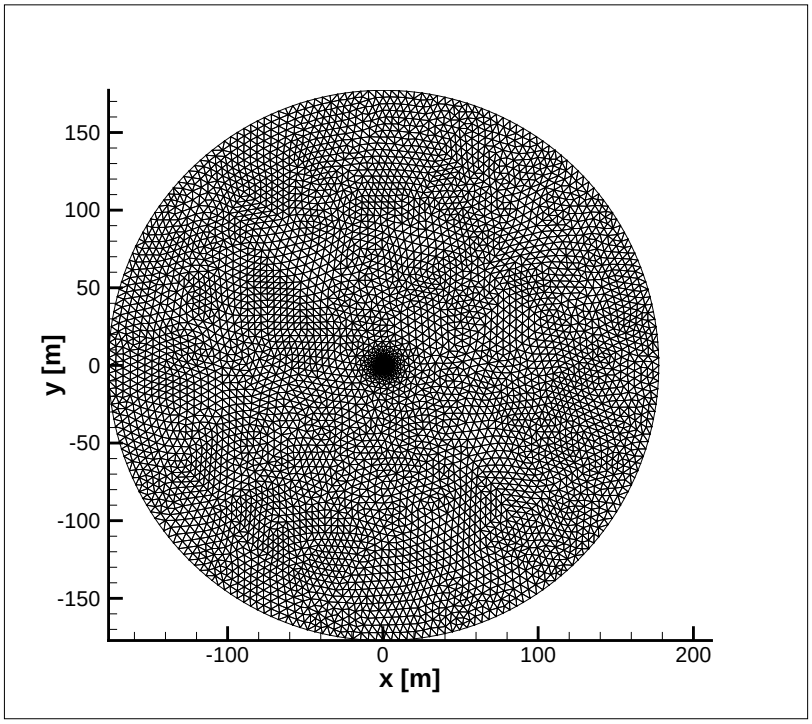


Figure 4.3: Flow domain with mesh for the NACA0012 test case

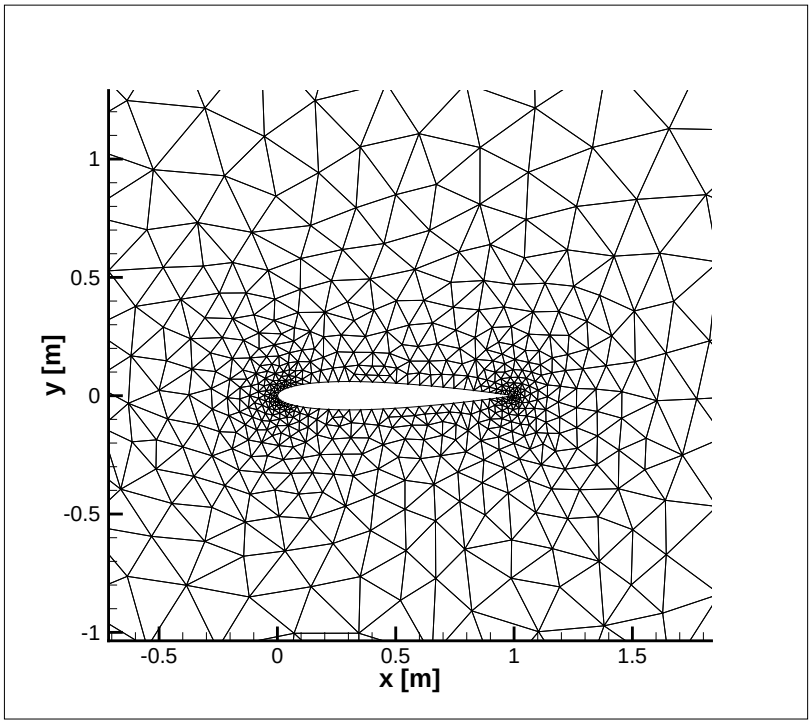
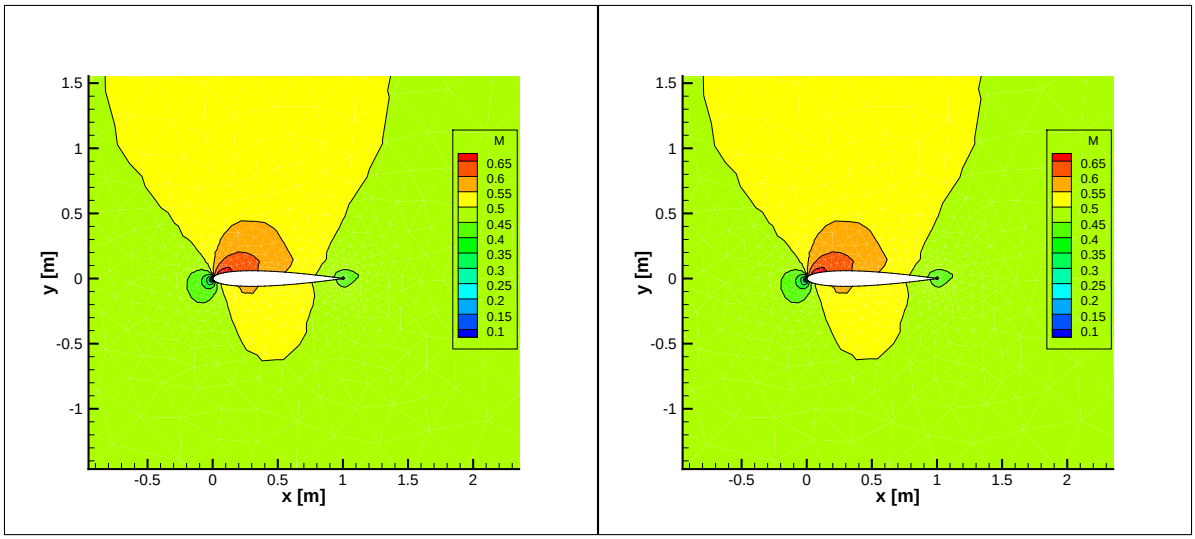
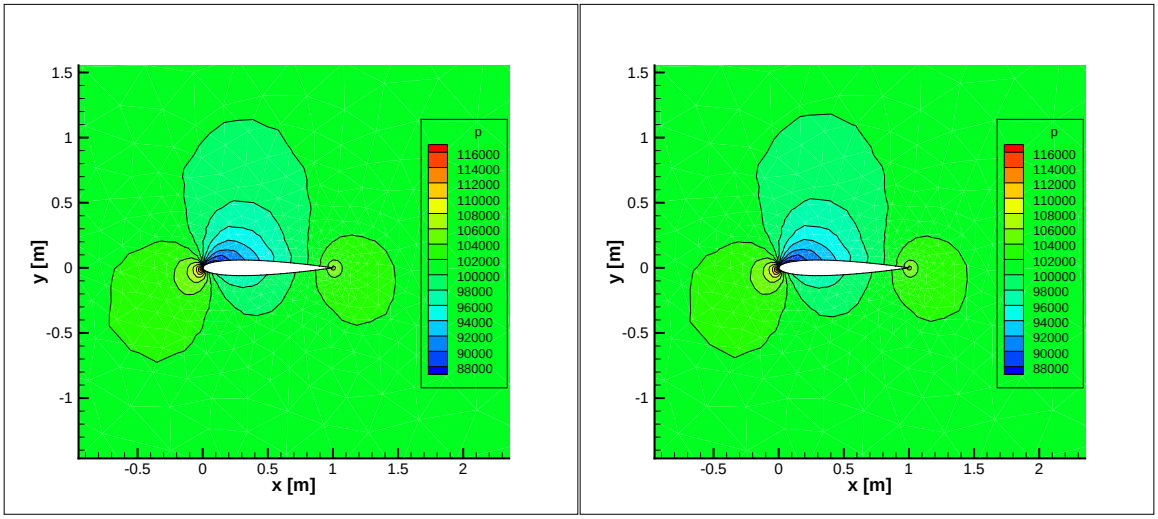


Figure 4.4: Near field mesh for the NACA0012 test case

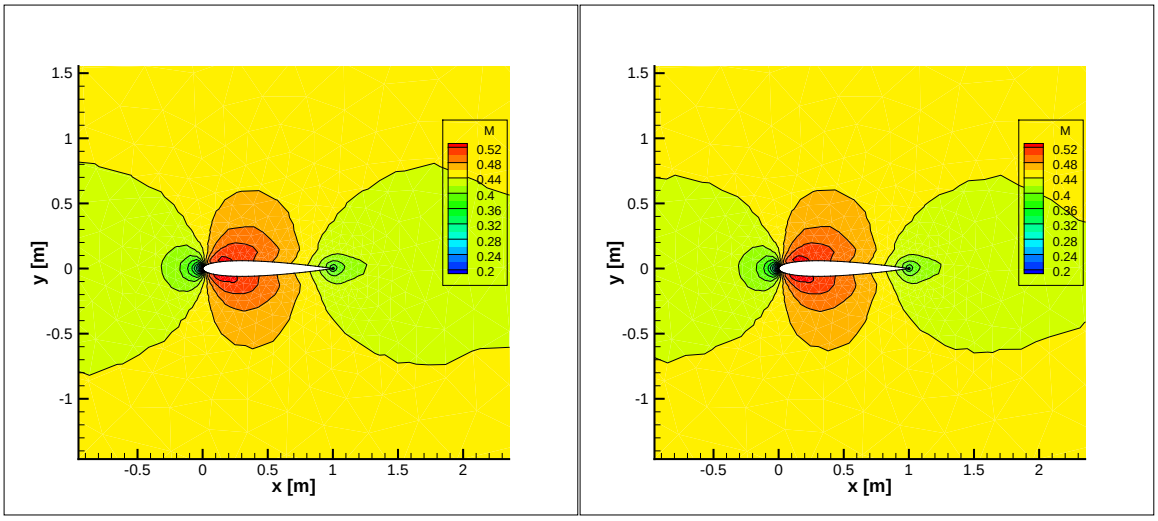


(a) Val. Case - 1. Mach Contours

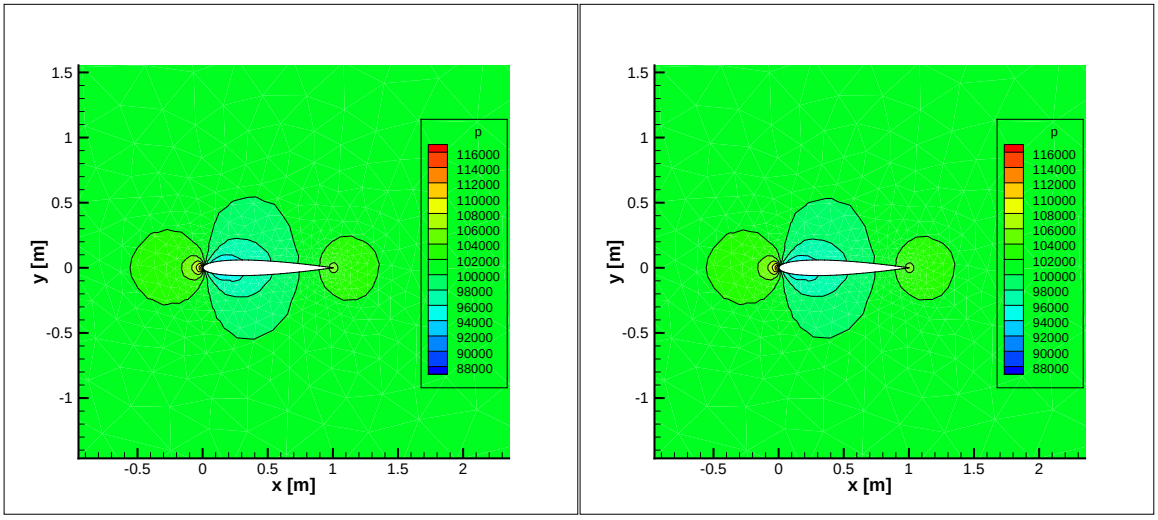


(b) Val. Case - 1. Pressure Contours

Figure 4.5: Comparison of the ROM predictions (right) to the FOM solution for validation case - 1



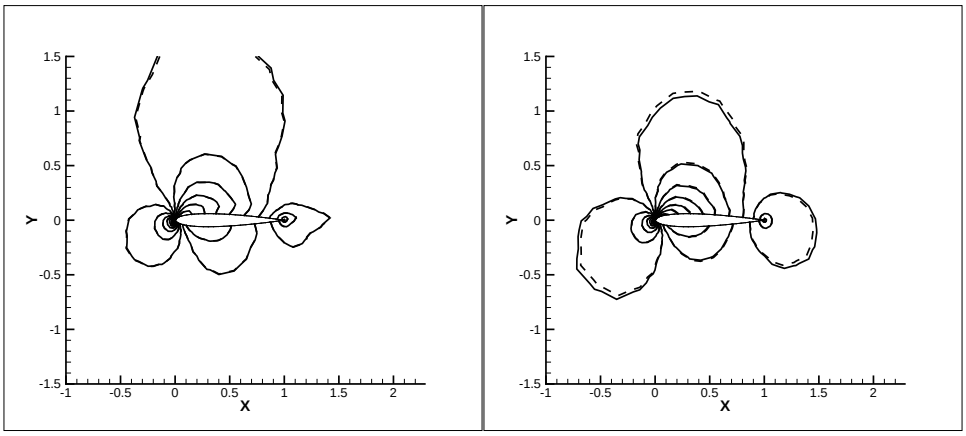
(a) Val. Case - 5. Mach Contours



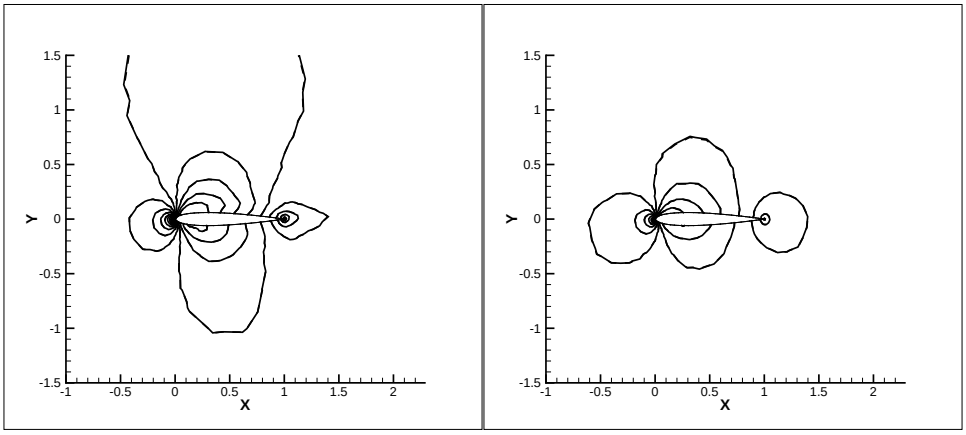
(b) Val. Case - 5. Pressure Contours

Figure 4.6: Comparison of the ROM predictions (right) to the FOM solution for validation case - 5

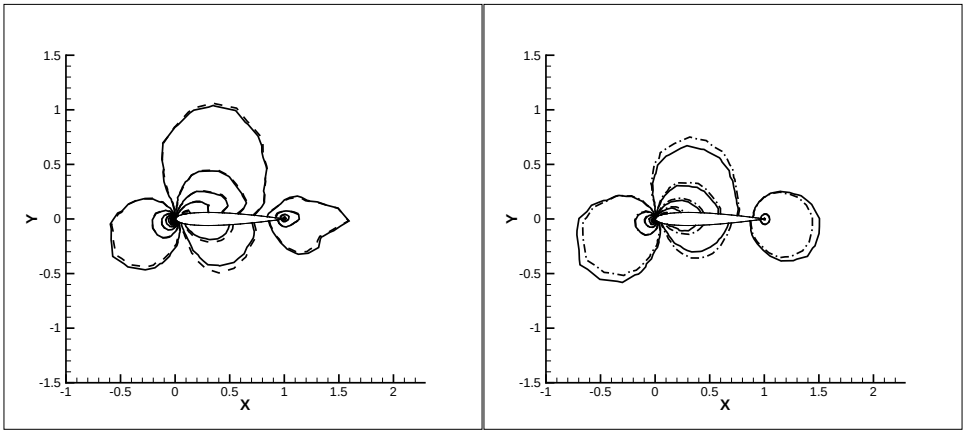
terms of capturing the non-linear flow field. The accuracy of the ROM serves as a proof again that the current formulation satisfies governing equations at the ROM level, thereby serving as a physics-based surrogate model. Additionally, the ROM evaluates at a wall clock time in $\mathcal{O}(1)\ sec$ giving 2-3 orders of magnitude speedup compared to the FOM on a desktop computer, offering the suitability to be used in a design optimization framework. While the present test case was specifically chosen to test the method under a shock-free flow regime, in the following test case the Mach number range is expanded.



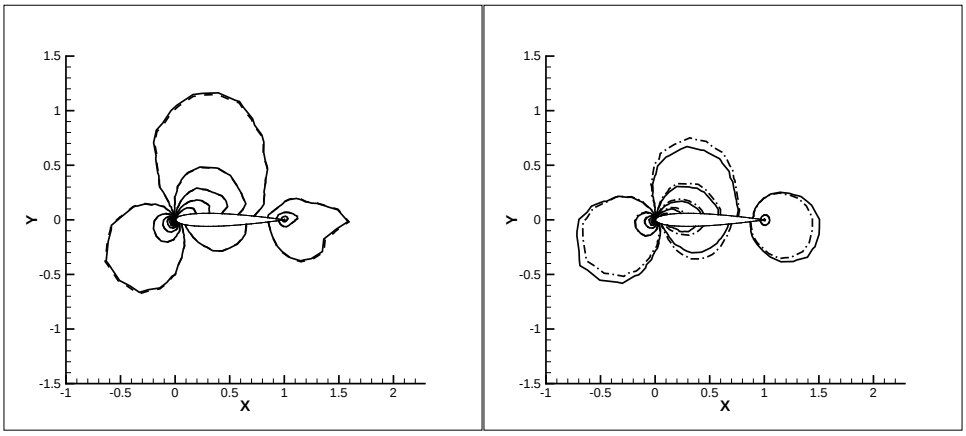
(a) $\mathbb{M} = 0.51$ $\alpha = 1.77$ $deg.$



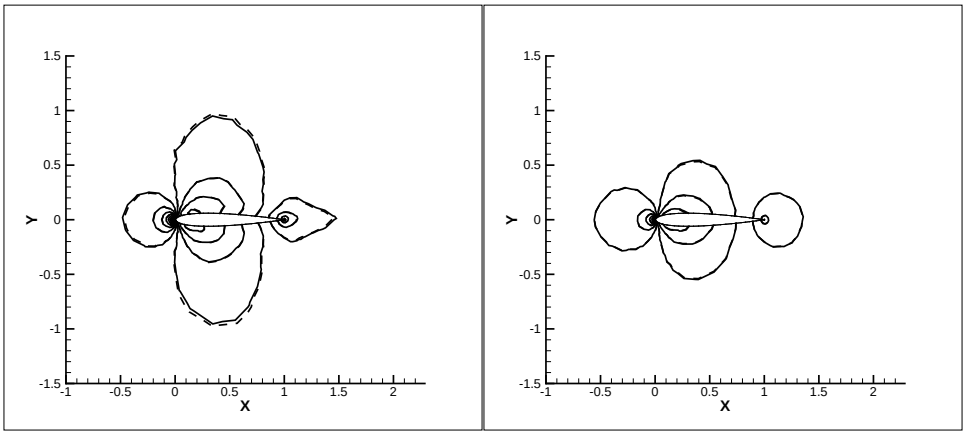
(b) $\mathbb{M} = 0.477$ $\alpha = 0.82$ $deg.$



(c) $\mathbb{M} = 0.32$ $\alpha = 1.36$ $deg.$



(d) $\mathbb{M} = 0.44$ $\alpha = 2.18$ $deg.$



(e) $\mathbb{M} = 0.46$ $\alpha = 0$ $deg.$

Figure 4.7: Comparison of ROM (dashed lines) and FOM (solid lines) for the NACA0012 validation cases. In each figure, left=Mach contours, right=Pressure contours

Table 4.3: Free-stream conditions for the RAE2822 test case

P_∞	28,745 Pa
ρ_∞	0.44 kg/m^3
a_∞	301.86 m/s
μ_∞	1.49E-5 $Pa-s$

4.1.2 RAE2822

Research Question 3. *How does the ROM perform in terms of system outputs?*

The RAE2822 airfoil shape is asymmetric as shown in Figure 4.8, whose coordinates are extracted from [132] and is used as the second test case. The flow domain is a circle of approximately 100 chord lengths, similar to the NACA test case and is discretized with a mesh of polyhedral cells, as shown in Figure 4.9. The mesh consists of 27,857 polyhedral cells each of which contains 4-8 faces and 5,777 vertices. The near-field mesh is made finer (Figure 4.10) in order to resolve the shocks reasonably well.

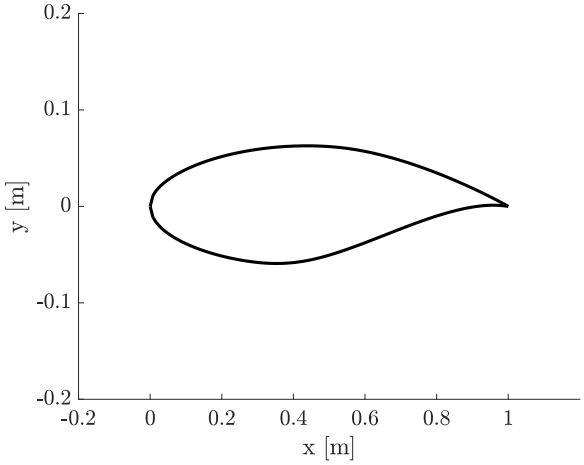


Figure 4.8: The RAE2822 airfoil shape

Predominantly shock-free parameter space

The parameter ranges are chosen such that the flow is predominantly shock-free, while shocks can still be expected as we approach toward the corner of the design space. The

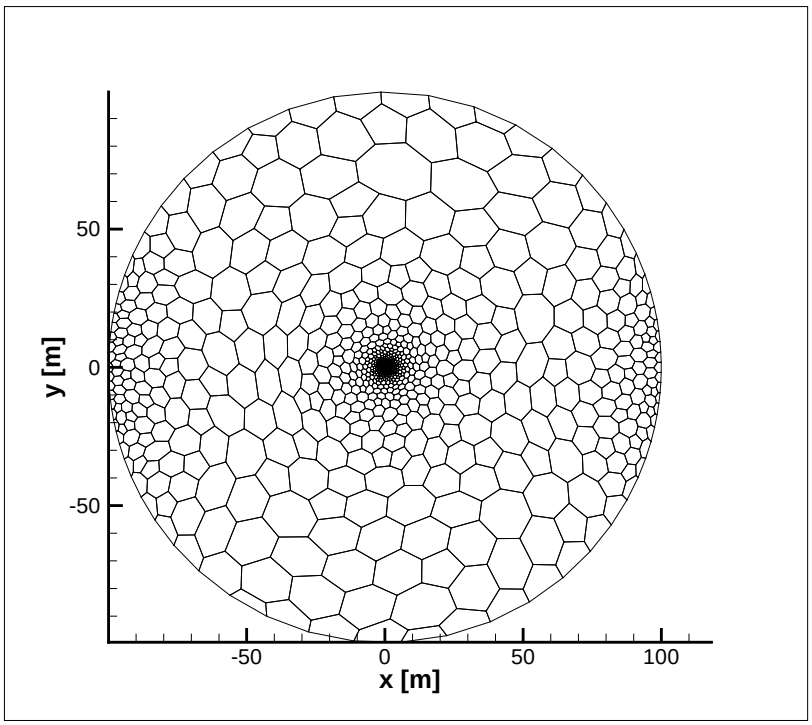


Figure 4.9: Flow domain with mesh for the RAE2822 test case

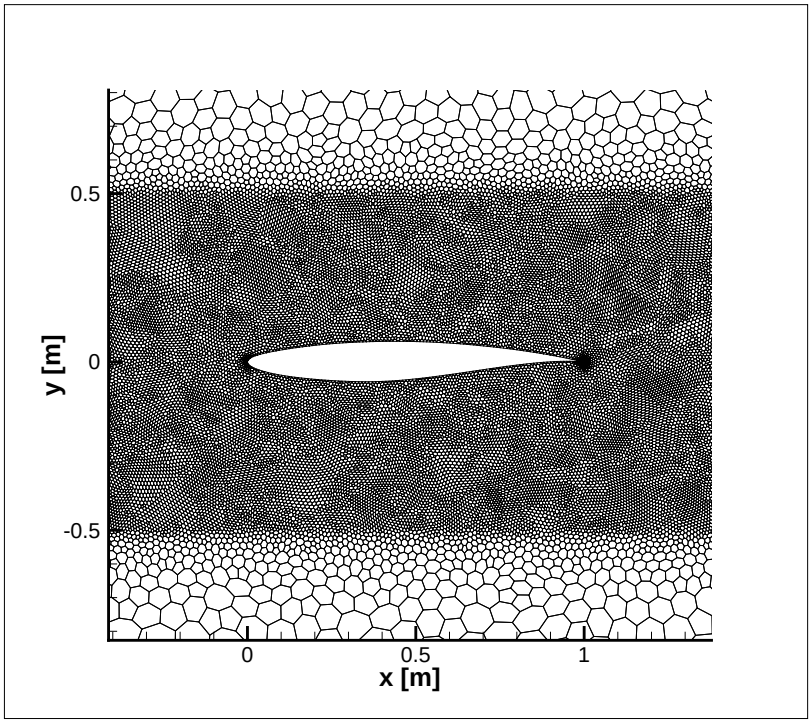


Figure 4.10: Near field mesh for the RAE2822 test case

Table 4.4: Output prediction error for the subsonic RAE-2822 test case

Case	C_P Error %	C_l (ROM)	C_l (FOM)	Error %
1	4.18	0.4059	0.4232	4.09
2	1.04	0.5445	0.5459	0.26
3	0.06	0.5229	0.5300	0.02
4	0.17	0.3375	0.3378	0.09
5	3.80	0.7183	0.7218	0.48
6	1.36	0.7090	0.7096	0.08
7	0.04	0.3744	0.3744	0.00
8	4.15	0.8385	0.8386	0.01
9	1.44	0.5998	0.5988	0.17
10	2.17	0.7181	0.7174	0.10

ranges are set as $\mathbb{M} \in [0.5, 0.7]$ and $\alpha \in [0, 3]deg.$. The free-stream conditions are summarized in Table 4.3. The snapshot locations in parameter space is shown in Figure 4.11 where the shaded circles represent the validation points and the rest are used for training the ROM. A total of 80 training points are used in this test case. Sample flow snapshots of the pressure and mach number are shown in Figure 4.12, that demonstrate that the design space could contain contrasting flow features.

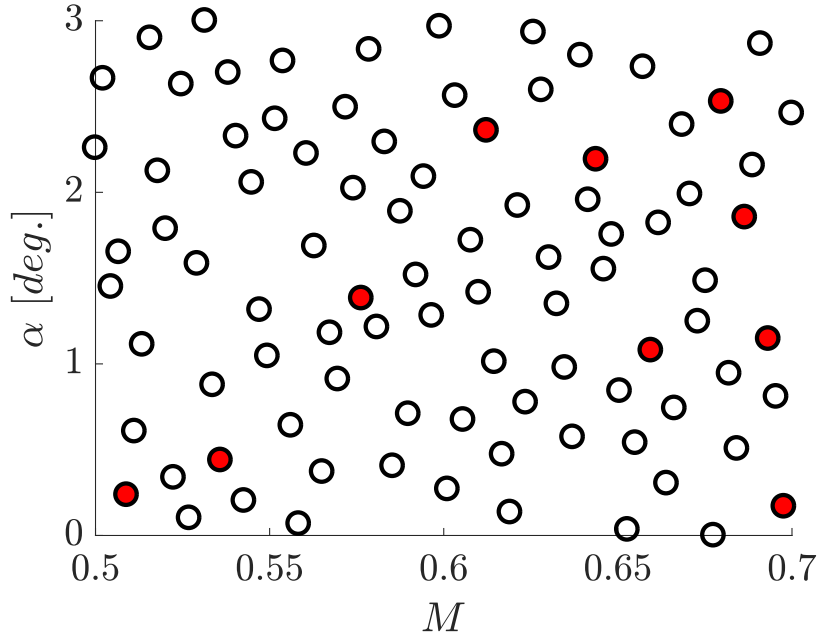
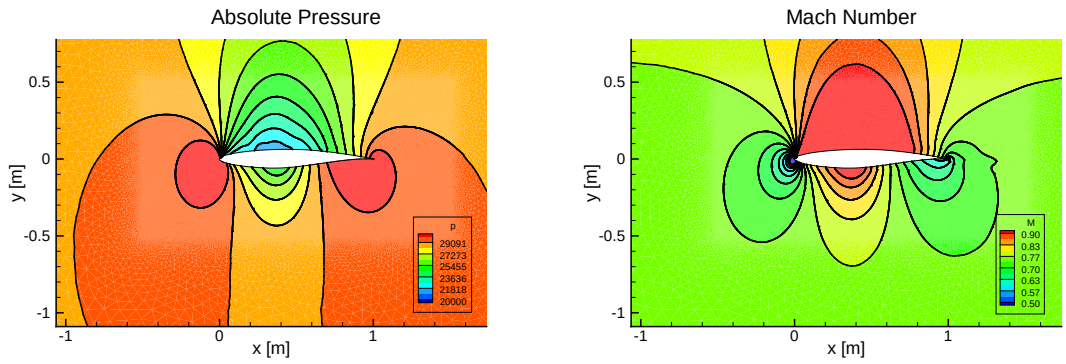
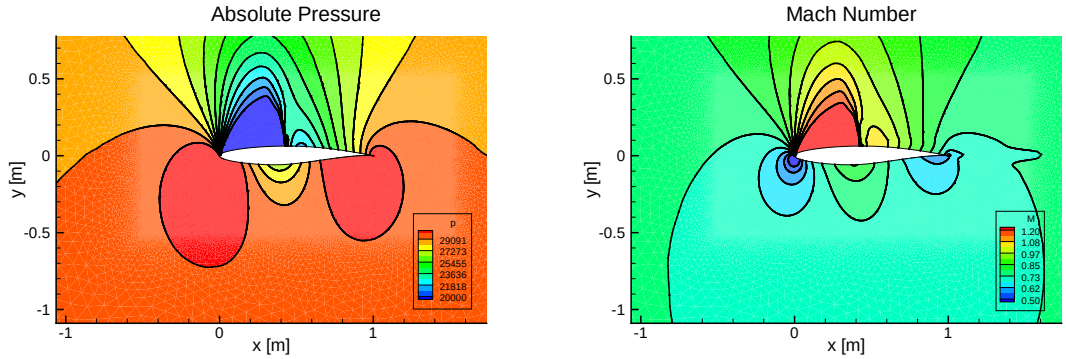


Figure 4.11: Parameter snapshots for the RAE2822 test case

The Mach number and pressure contour predictions are shown in Figure 4.14 overlaid



(a) $\mathbb{M} = 0.7$, $\alpha = 0.5148\ deg.$

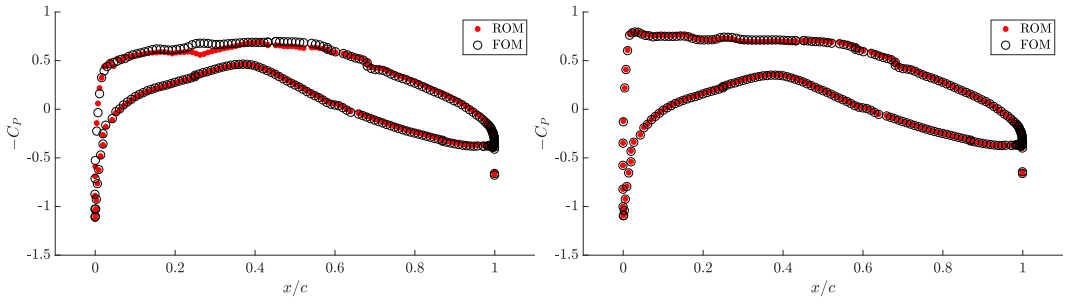


(b) $\mathbb{M} = 0.6953$, $\alpha = 2.8225\ deg.$

Figure 4.12: Flow snapshots from the $\mathbb{M} \in [0.5, 0.7]$ and $\alpha \in [0, 3]\ deg.$ parameter range. 2 contrasting snapshots with and without shocks are shown. The flow regime however is predominantly shock-free.

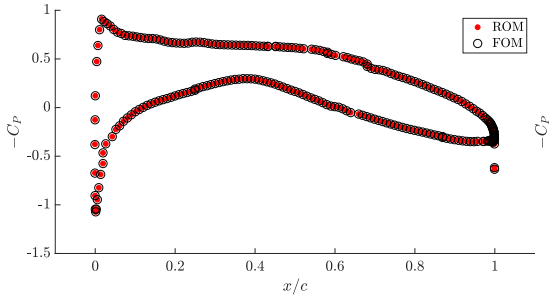
with the true FOM solution. The ability of the present approach to accurately capture the field variables is emphasized. Specifically, the method is able to accurately distinguish between shock-free parameter combinations from those with shocks. The pressure coefficient comparison is shown in Figure 4.13 and the associated error in the outputs are summarized in Table 4.4. The C_P is predicted with an average error of $\approx 2\%$ and a maximum of 5% across all validation cases. The error for lift coefficient is also summarized in the table and notice that it is predicted with similar accuracy as C_P. The drag coefficient comparison is not shown because under inviscid flow conditions at subsonic mach numbers, the drag coefficient is very small (0-10 counts) and also is prone to be contaminated with noise that is an artifact of the numerics and grid resolutions. Under such conditions, the error

computation for drag is meaningless since it can lead to very high discrepancies. However, this is not true for the C_l and its predictions are comparable to that of C_P, as shown in the results. Finally, the ROM evaluates at a wall-clock time of roughly 1 seconds compared to the FOM which takes about 600 seconds under identical circumstances (serial mode execution, convergence tolerance of 10^{-6} on all residuals). Overall the ROM is able to predict both the state and outputs within 5% accuracy at a fraction of the cost of the FOM.

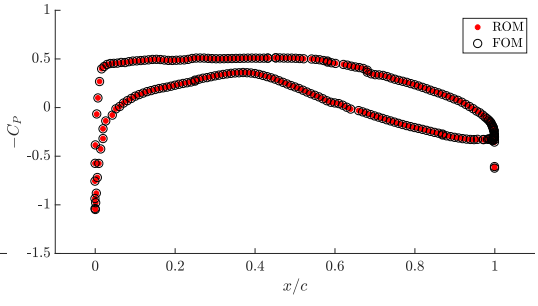


(a) $[\mathbb{M}, \alpha] = [0.70, 0.17]$

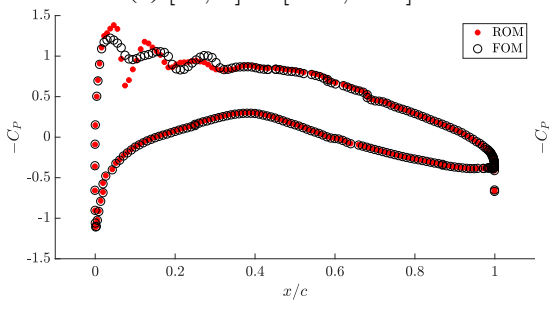
(b) $[\mathbb{M}, \alpha] = [0.66, 1.08]$



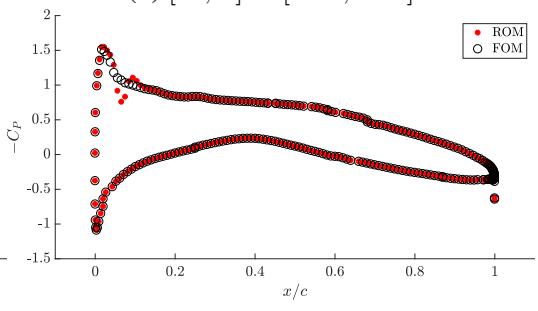
(c) $[\mathbb{M}, \alpha] = [0.58, 1.38]$



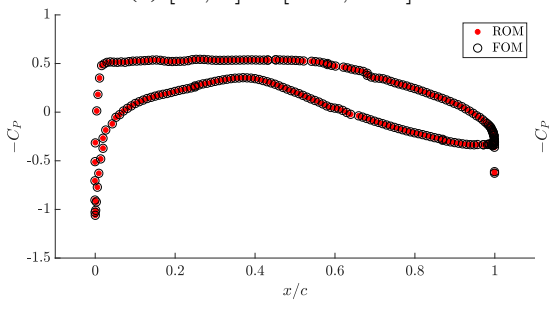
(d) $[\mathbb{M}, \alpha] = [0.51, 0.24]$



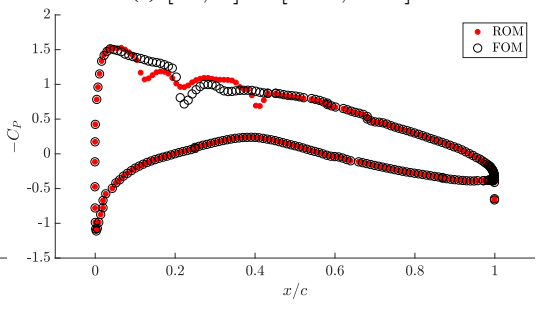
(e) $[\mathbb{M}, \alpha] = [0.69, 1.85]$



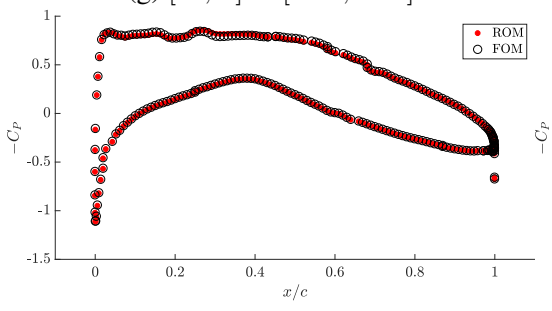
(f) $[\mathbb{M}, \alpha] = [0.61, 2.36]$



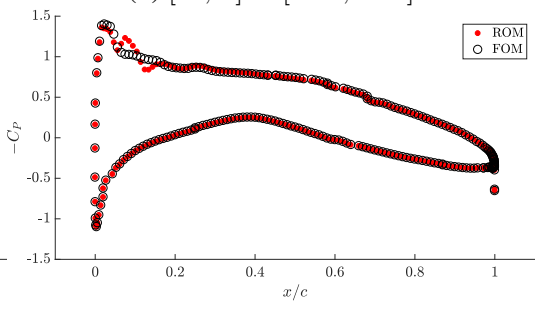
(g) $[\mathbb{M}, \alpha] = [0.54, 0.44]$



(h) $[\mathbb{M}, \alpha] = [0.68, 2.53]$

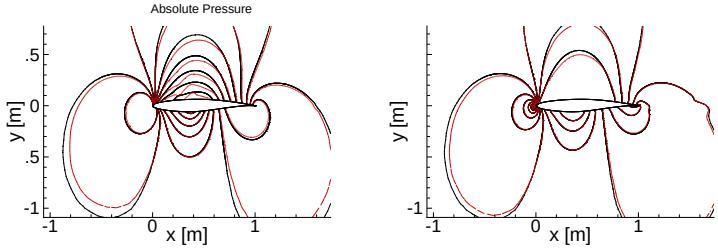


(i) $[\mathbb{M}, \alpha] = [0.69, 1.15]$

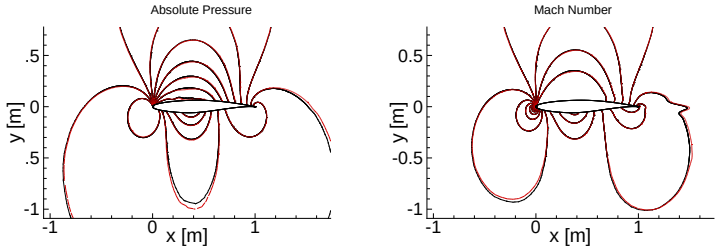


(j) $[\mathbb{M}, \alpha] = [0.64, 2.19]$

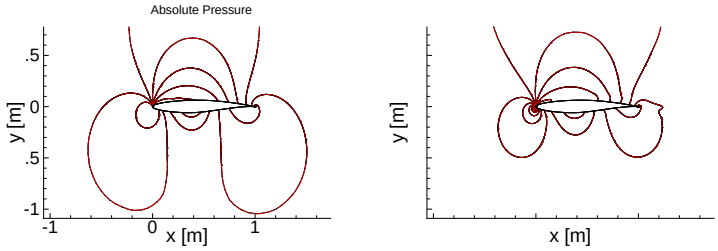
Figure 4.13: C_P comparison for the subsonic RAE2822 test case



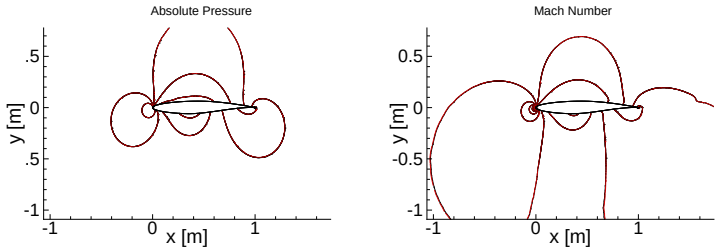
(a) $[\mathbb{M}, \alpha] = [0.70, 0.17]$



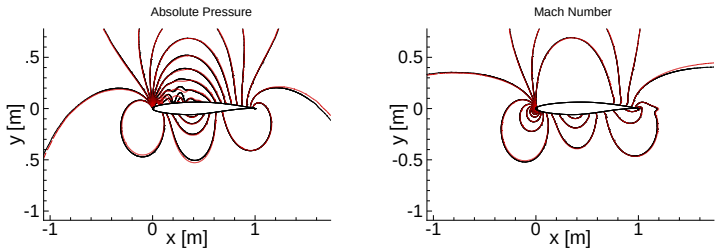
(b) $[\mathbb{M}, \alpha] = [0.66, 1.08]$



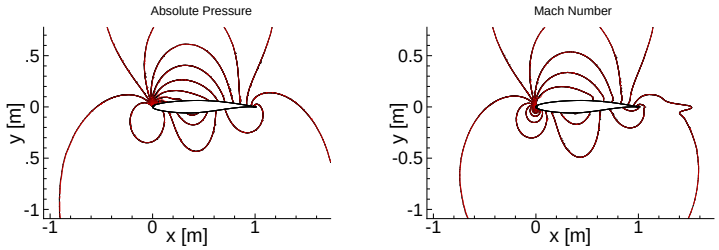
(c) $[\mathbb{M}, \alpha] = [0.58, 1.38]$



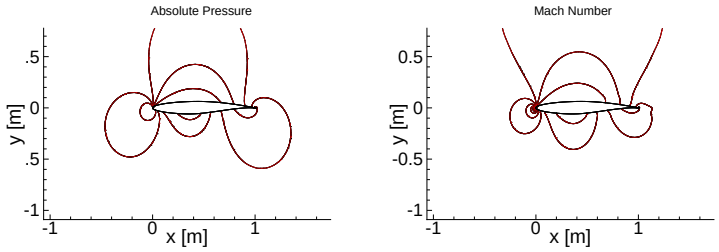
(d) $[\mathbb{M}, \alpha] = [0.51, 0.24]$



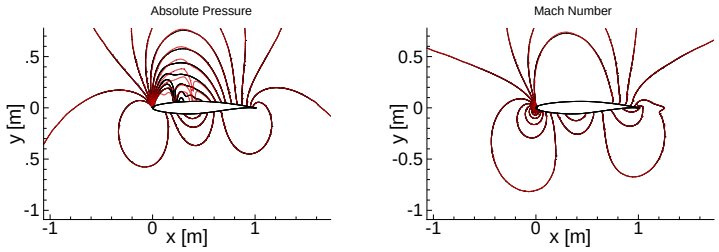
(e) $[\mathbb{M}, \alpha] = [0.69, 1.85]$



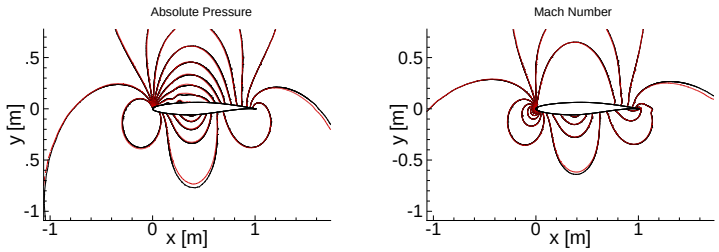
(f) $[\mathbb{M}, \alpha] = [0.61, 2.36]$



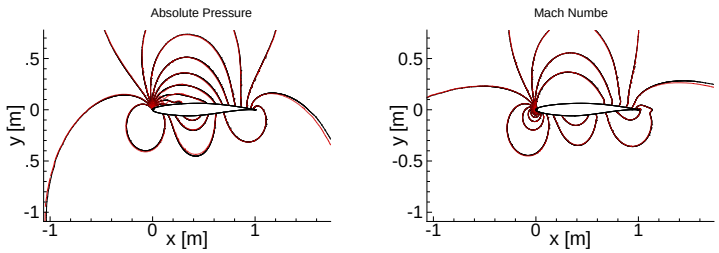
(g) $[\mathbb{M}, \alpha] = [0.54, 0.44]$



(h) $[\mathbb{M}, \alpha] = [0.68, 2.53]$



(i) $[\mathbb{M}, \alpha] = [0.69, 1.15]$



(j) $[\mathbb{M}, \alpha] = [0.64, 2.19]$

Figure 4.14: Absolute pressure and mach number contours for the subsonic RAE2822 test case. Red lines = ROM, Black lines = FOM.

Table 4.5: Training set size and parameter ranges for the RAE2822 transonic test case

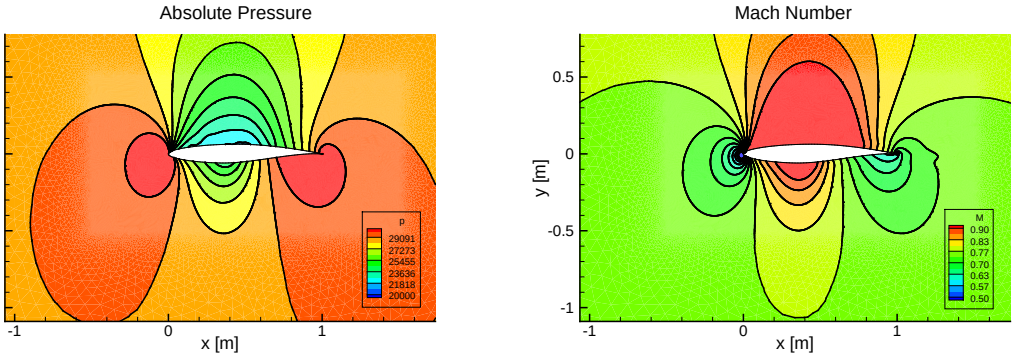
DOE	Training Snapshots	Mach Range	α Range
1	40	$[0.7,\ 0.9]$	$[0,\ 3]$
2	80	$[0.7,\ 0.9]$	$[0,\ 3]$
3	120	$[0.7,\ 0.9]$	$[0,\ 3]$
4	80	$[0.8,\ 0.9]$	$[0,\ 2]$

Transonic mach number regime

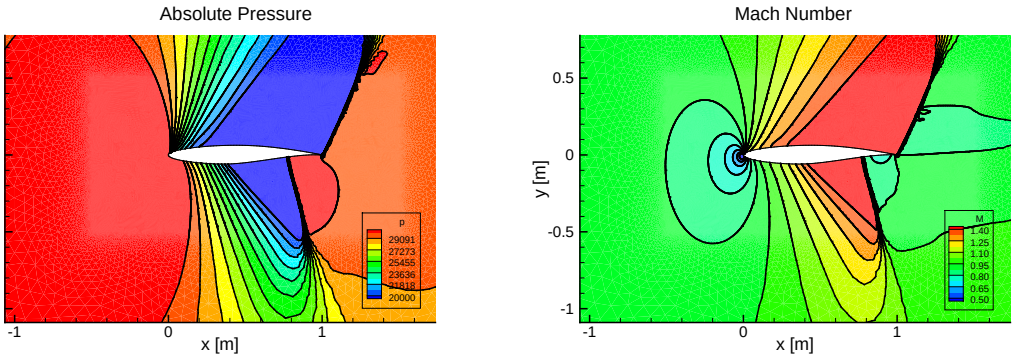
Research Question 4. *(a) How does the ROM perform in the presence of moving shocks? (b) How many snapshots are required to accurately predict outputs?*

The parameter ranges are now chosen such that the flow almost always contains a shock at some location along the airfoil. The ranges are set as $\mathbb{M} \in [0.7, 0.9]$ and $\alpha \in [0, 3]deg.$. Note that within this parameter range, there are also some shock-free snapshots such as the one shown in Figure 4.15a; therefore this is also a test for the method to appropriately distinguish between such flows when used for prediction. We test the method with various densities of distribution of the snapshots. The test cases are summarized in Table 4.5 and the corresponding snapshot points are shown in Figures 4.16.

The pressure coefficient plots for each case are shown in Figures 4.17 through 4.20. With a small training set (DOE-1), the inability of the ROM to accurately predict the shock location and strength is evident. The discrepancy is particularly large at the location of the shock. We further increase the training set size to 80 and 120 in the DOE-2 and DOE-3 respectively, to observe that the predictions improve, yet incorrectly predicting the shock location in a few test cases. We observe that with multiple shocks and with greater variation in the shock location, the ROM does not capture the pressure distributions exactly. Therefore, we go one step further and shrink the parameter ranges ($\mathbb{M} \in [0.8, 0.9],\ \alpha \in [0, 2]$) in DOE-4 and observe that the prediction improves. The ROM predicts the shock location with at most 5% error. Overall, with highly non-linear flow fields, such as with shocks, the methodology requires a relatively much higher density of training points.

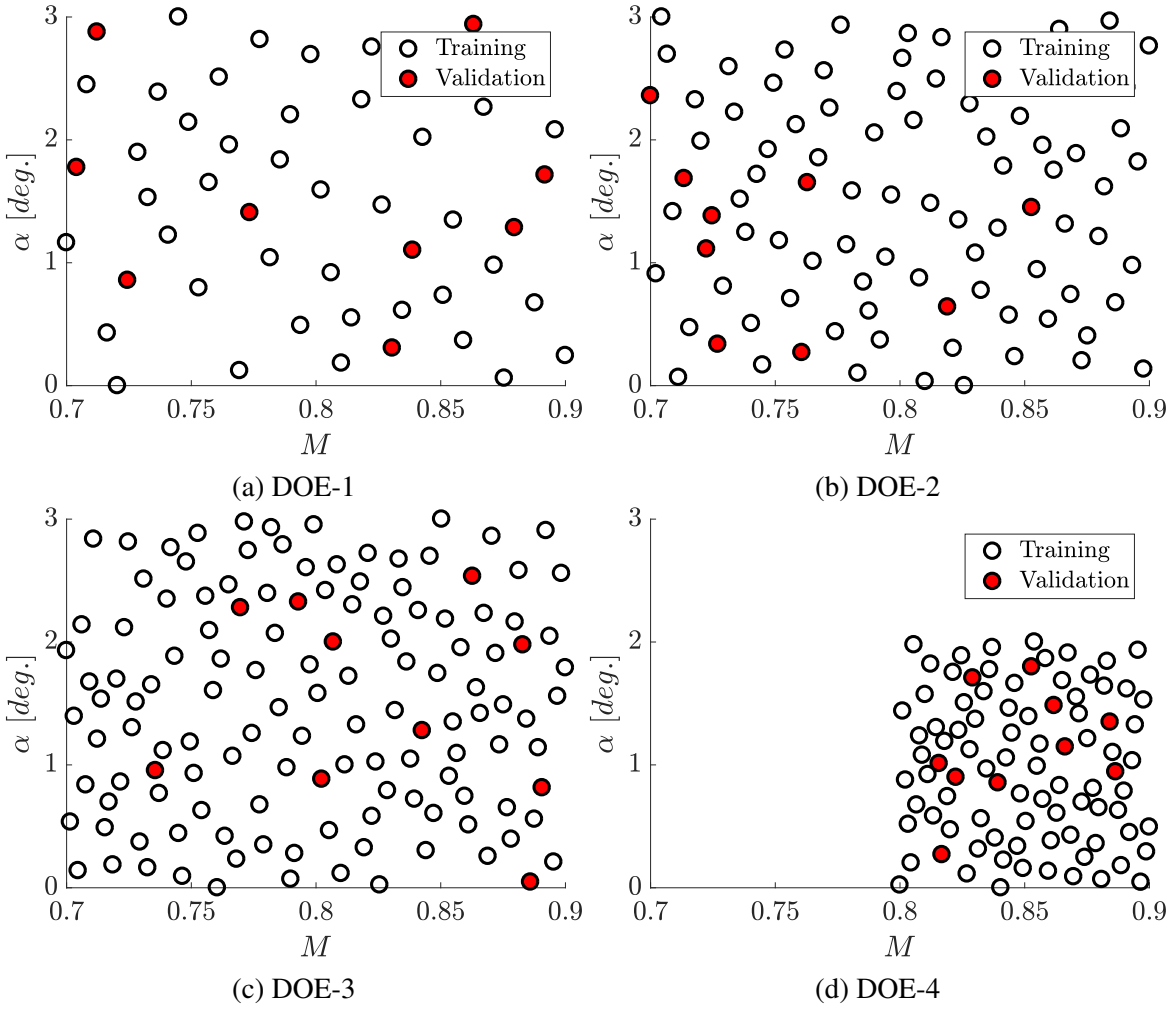


(a) $\mathbb{M} = 0.705$, $\alpha = 0.14\ deg.$



(b) $\mathbb{M} = 0.89$, $\alpha = 2.91\ deg.$

Figure 4.15: Flow snapshots from the $\mathbb{M} \in [0.7, 0.9]$ and $\alpha \in [0, 3]\ deg.$ parameter range. 2 contrasting snapshots with and without shock are shown. The flow regime however, almost always contains a shock.



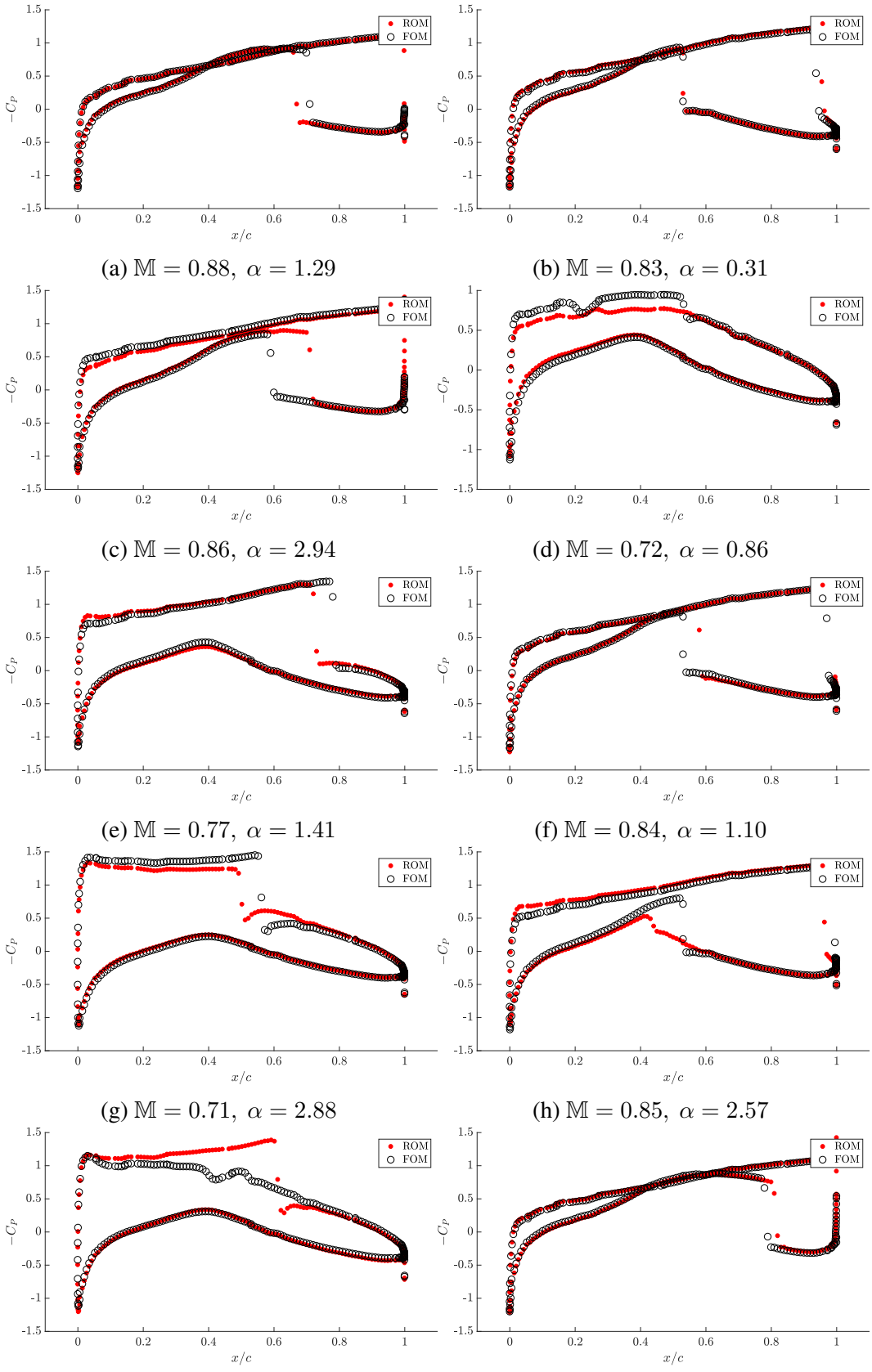
(a) DOE-1

(b) DOE-2

(c) DOE-3

(d) DOE-4

Figure 4.16: Snapshot locations for the transonic RAE2822 test case



(a) $\mathbb{M} = 0.88,\ \alpha = 1.29$

(b) $\mathbb{M} = 0.83,\ \alpha = 0.31$

(c) $\mathbb{M} = 0.86,\ \alpha = 2.94$

(d) $\mathbb{M} = 0.72,\ \alpha = 0.86$

(e) $\mathbb{M} = 0.77,\ \alpha = 1.41$

(f) $\mathbb{M} = 0.84,\ \alpha = 1.10$

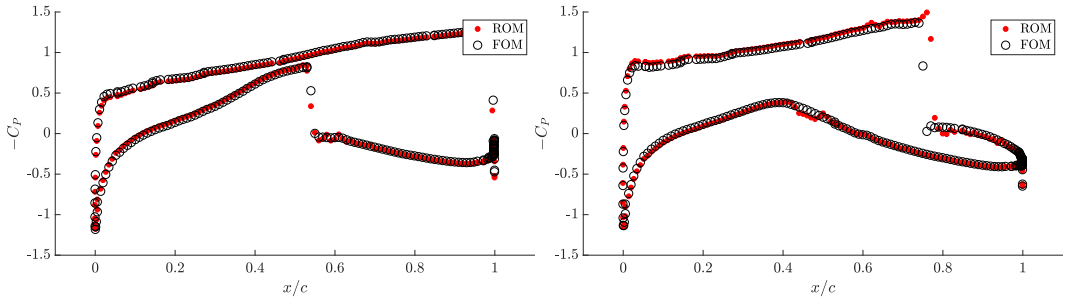
(g) $\mathbb{M} = 0.71,\ \alpha = 2.88$

(h) $\mathbb{M} = 0.85,\ \alpha = 2.57$

(i) $\mathbb{M} = 0.70,\ \alpha = 1.78$

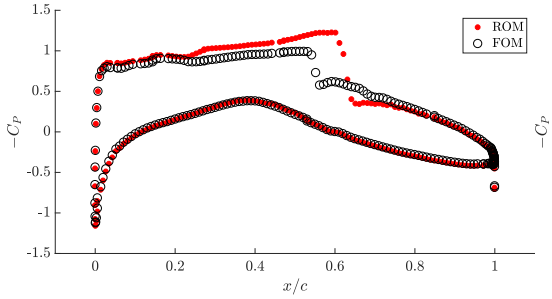
(j) $\mathbb{M} = 0.89,\ \alpha = 1.71$

Figure 4.17: C_P comparison for the transonic RAE2822 test case (DOE-1)

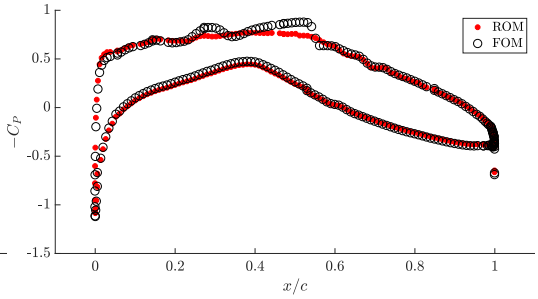


(a) $\mathbb{M} = 0.85,\ \alpha = 2.53$

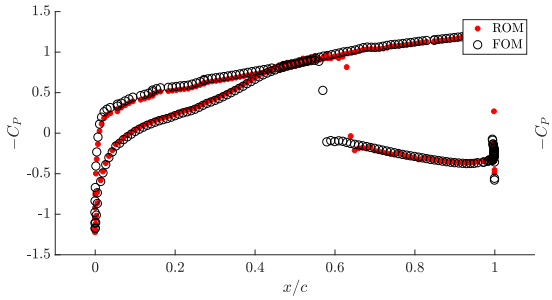
(b) $\mathbb{M} = 0.76,\ \alpha = 1.65$



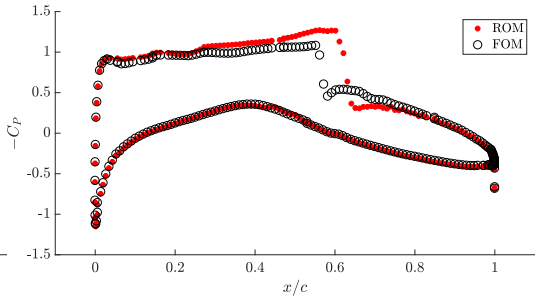
(c) $\mathbb{M} = 0.72,\ \alpha = 1.11$



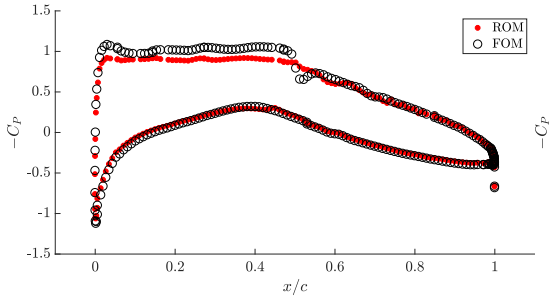
(d) $\mathbb{M} = 0.73,\ \alpha = 0.34$



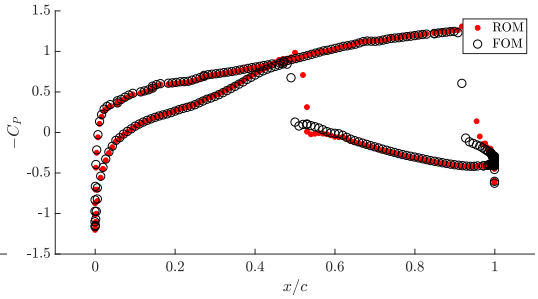
(e) $\mathbb{M} = 0.85,\ \alpha = 1.45$



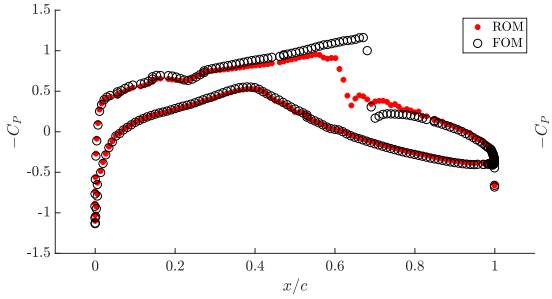
(f) $\mathbb{M} = 0.73,\ \alpha = 1.38$



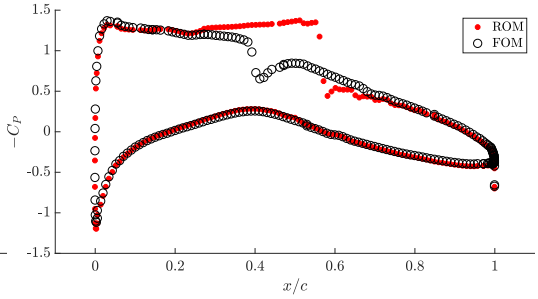
(g) $\mathbb{M} = 0.71,\ \alpha = 1.69$



(h) $\mathbb{M} = 0.82,\ \alpha = 0.64$

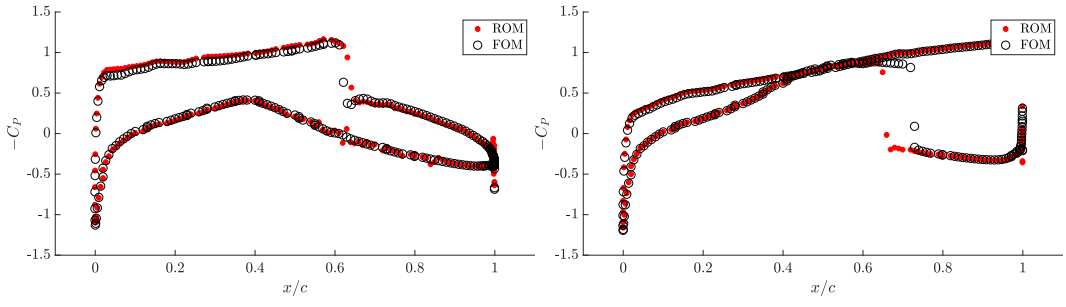


(i) $\mathbb{M} = 0.76,\ \alpha = 0.27$



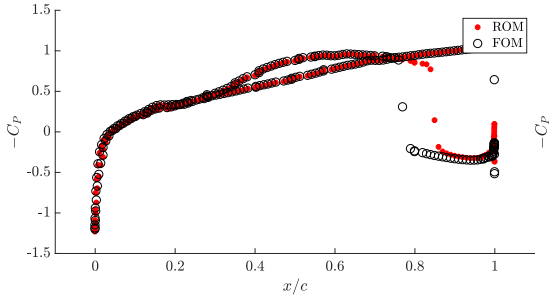
(j) $\mathbb{M} = 0.70,\ \alpha = 2.36$

Figure 4.18: C_P comparison for the transonic RAE2822 test case (DOE-2)

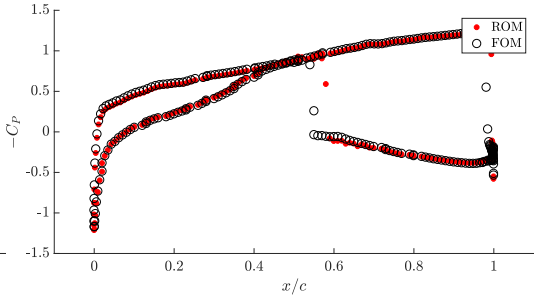


(a) $\mathbb{M} = 0.74,\ \alpha = 0.95$

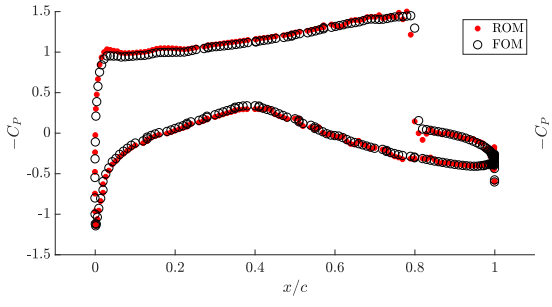
(b) $\mathbb{M} = 0.88,\ \alpha = 1.98$



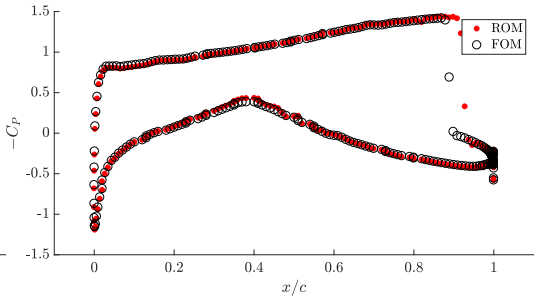
(c) $\mathbb{M} = 0.89,\ \alpha = 0.05$



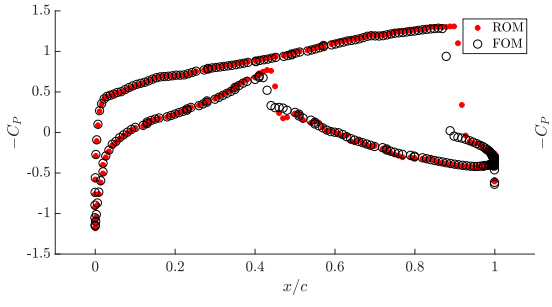
(d) $\mathbb{M} = 0.84,\ \alpha = 1.28$



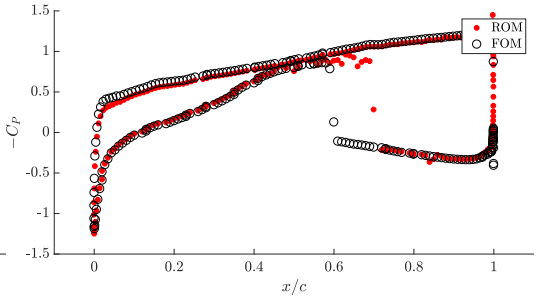
(e) $\mathbb{M} = 0.77,\ \alpha = 2.28$



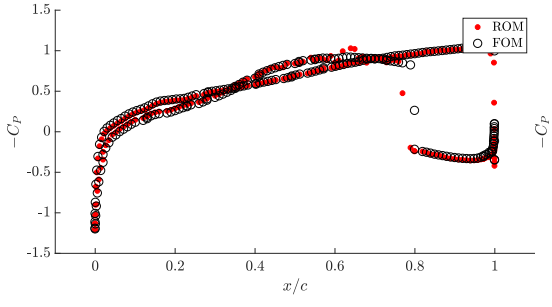
(f) $\mathbb{M} = 0.79,\ \alpha = 2.33$



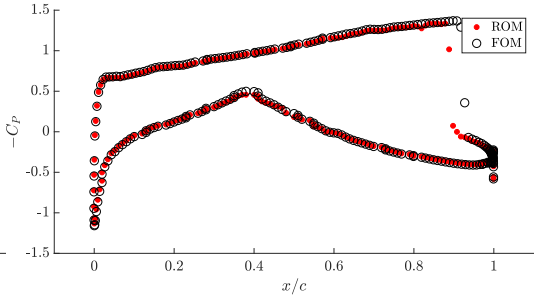
(g) $\mathbb{M} = 0.80,\ \alpha = 0.88$



(h) $\mathbb{M} = 0.86,\ \alpha = 2.53$

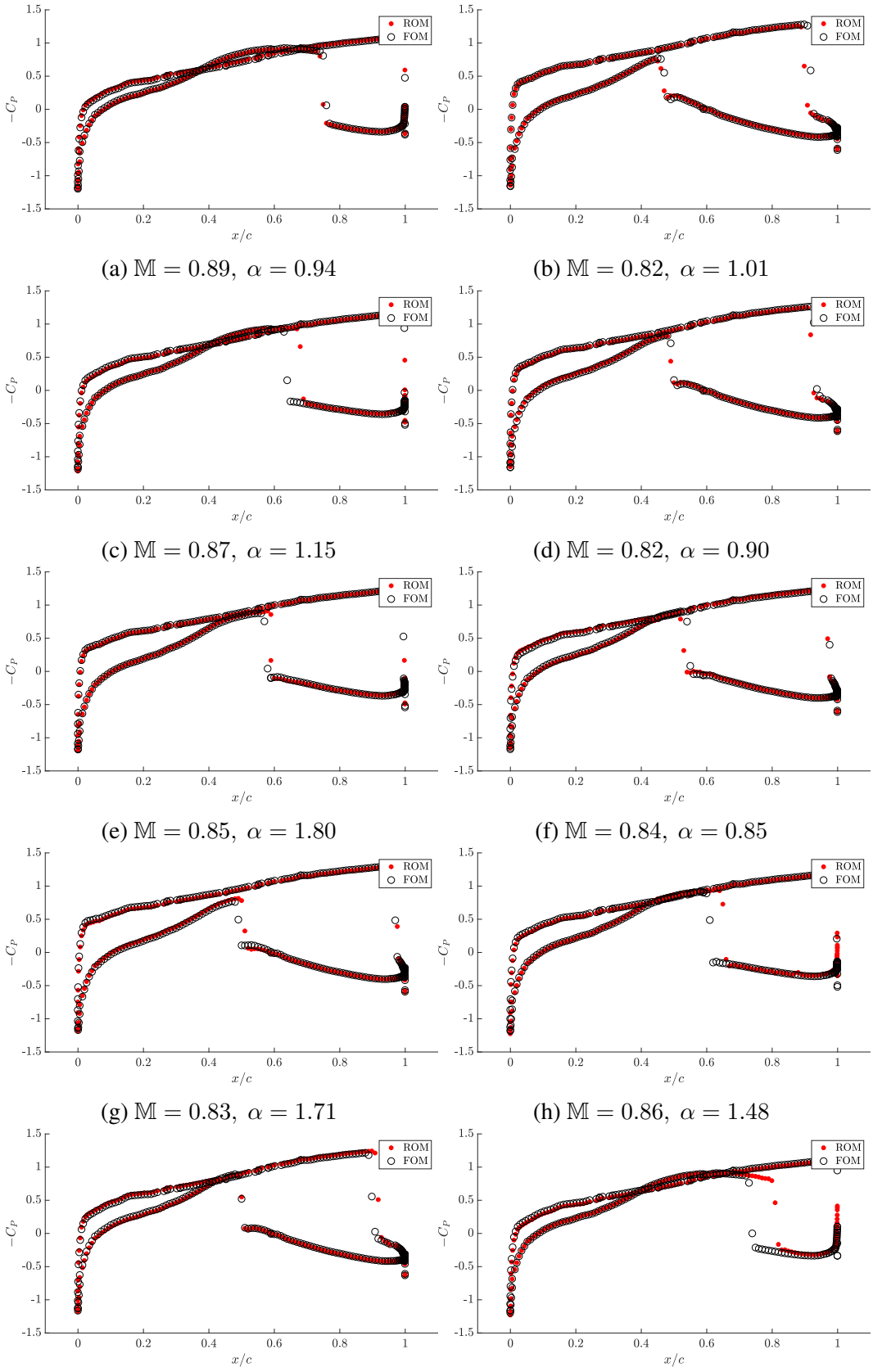


(i) $\mathbb{M} = 0.89,\ \alpha = 0.81$



(j) $\mathbb{M} = 0.81,\ \alpha = 2.00$

Figure 4.19: C_P comparison for the transonic RAE2822 test case (DOE-3)



(a) $\mathbb{M} = 0.89,\ \alpha = 0.94$

(b) $\mathbb{M} = 0.82,\ \alpha = 1.01$

(c) $\mathbb{M} = 0.87,\ \alpha = 1.15$

(d) $\mathbb{M} = 0.82,\ \alpha = 0.90$

(e) $\mathbb{M} = 0.85,\ \alpha = 1.80$

(f) $\mathbb{M} = 0.84,\ \alpha = 0.85$

(g) $\mathbb{M} = 0.83,\ \alpha = 1.71$

(h) $\mathbb{M} = 0.86,\ \alpha = 1.48$

(i) $\mathbb{M} = 0.82,\ \alpha = 0.27$

(j) $\mathbb{M} = 0.88,\ \alpha = 1.35$

Figure 4.20: C_P comparison for the transonic RAE2822 test case (DOE-4)

Table 4.6: Comparison of C_p, C_l and C_d predicted by ROM against the FOM for the transonic RAE2822 test case (DOE-1)

Case	C_P Error %	C_d (ROM)	C_d (FOM)	Error %	C_l (ROM)	C_l (FOM)	Error %
1	5.47	0.0993	0.1071	7.28	0.4861	0.4562	6.55
2	2.86	0.0641	0.0585	9.57	0.6779	0.6210	9.16
3	11.17	0.1322	0.1202	9.98	0.5938	0.7937	25.18
4	5.48	0.0054	0.0027	100	0.4832	0.5971	19.08
5	6.02	0.0178	0.0309	42.39	0.7935	0.8133	2.44
6	8.61	0.0931	0.0785	18.60	0.7045	0.7370	4.41
7	6.99	0.0122	0.0121	0.83	0.8892	0.9916	10.33
8	13.53	0.0963	0.1101	12.53	0.9791	0.8855	10.57
9	8.96	0.0066	0.0032	106.25	0.8838	0.7415	19.19
10	4.21	0.1228	0.1187	3.45	0.3759	0.3778	0.50

Table 4.7: Comparison of C_p, C_l and C_d predicted by ROM against the FOM for the transonic RAE2822 test case (DOE-2)

Case	C_P Error %	C_d (ROM)	C_d (FOM)	Error %	C_l (ROM)	C_l (FOM)	Error %
1	3.85	0.1076	0.1102	2.36	0.8257	0.8535	3.26
2	3.41	0.0313	0.0268	16.79	0.9088	0.8434	7.75
3	5.55	0.0044	0.003	46.67	0.7354	0.6452	13.98
4	2.48	0.0007	0.0024	70.83	0.4888	0.4924	0.73
5	9.11	0.1029	0.0928	10.88	0.6022	0.7072	14.85
6	4.77	0.006	0.0037	62.16	0.7793	0.7074	10.16
7	4.47	0.006	0.0034	76.47	0.6689	0.7472	10.48
8	6.80	0.0682	0.0548	24.45	0.7260	0.7060	2.83
9	4.87	0.0045	0.0077	41.56	0.4666	0.5300	11.96
10	7.12	0.0099	0.004	147.50	0.9426	0.8556	10.17

Table 4.8: Comparison of C_p, C_l and C_d predicted by ROM against the FOM for the transonic RAE2822 test case (DOE-3)

Case	C_P Error %	C_d (ROM)	C_d (FOM)	Error %	C_l (ROM)	C_l (FOM)	Error %
1	3.48	0.0027	0.0043	37.21	0.6801	0.6391	6.42
2	5.32	0.1067	0.1155	7.62	0.5861	0.5025	16.64
3	6.75	0.1168	0.1063	9.88	0.1548	0.234	33.85
4	7.52	0.0941	0.0833	12.97	0.6994	0.7368	5.08
5	3.58	0.0427	0.0464	7.97	1.0066	0.9814	2.57
6	3.28	0.0842	0.0714	17.93	1.0603	1.018	4.16
7	3.57	0.0552	0.0451	22.39	0.7956	0.751	5.94
8	9.98	0.1261	0.1135	11.10	0.6039	0.7549	20.00
9	5.84	0.1096	0.1125	2.58	0.2855	0.2841	0.49
10	2.87	0.0643	0.0758	15.17	0.9046	0.9604	5.81

Table 4.9: Comparison of C_p, C_l and C_d predicted by ROM against the FOM for the transonic RAE2822 test case (DOE-4)

Case	C_P Error %	C_d (ROM)	C_d (FOM)	Error %	C_l (ROM)	C_l (FOM)	Error %
1	1.47	0.1087	0.1099	1.09	0.3745	0.3446	8.68
2	1.70	0.0538	0.0594	9.43	0.7497	0.7787	3.72
3	5.18	0.1049	0.0966	8.59	0.4999	0.5619	11.03
4	1.64	0.0582	0.0624	6.73	0.7188	0.7483	3.94
5	3.99	0.1022	0.0982	4.07	0.7255	0.7506	3.34
6	2.42	0.0712	0.0748	4.81	0.7244	0.6968	3.96
7	2.38	0.0872	0.0844	3.32	0.8409	0.866	2.90
8	6.21	0.106	0.0979	8.27	0.5839	0.6411	8.92
9	2.31	0.0508	0.0458	10.92	0.6794	0.6272	8.32
10	5.63	0.1206	0.1109	8.75	0.3093	0.4162	25.68

4.1.3 Discussion

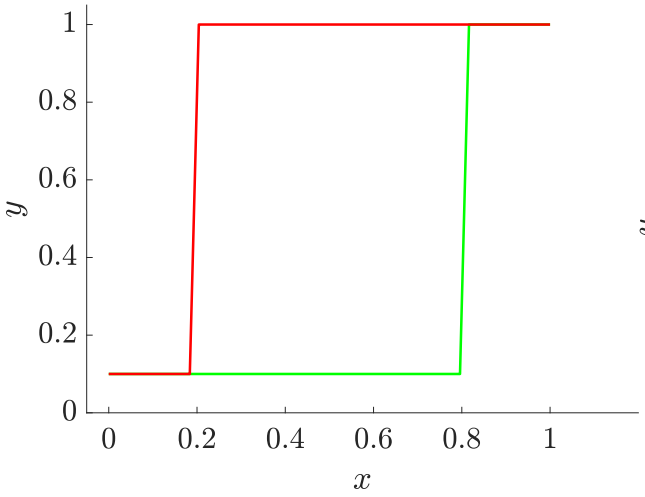
The results of the validation of the methodology demonstrates its capability to capture non-linear flow field with accuracy in $\mathcal{O}(1)\%$ at a computational cost that is negligible compared to that of the full-order high-fidelity model, specifically at subsonic shock-free flow conditions. The strength of the method is that the non-linearity in the system is effectively handled by introducing appropriate auxiliary equations, which serve as equality constraints to the optimization problem that represents the ROM (Eq.4.7).

Research Question 5. *(a) Is a POD-based approach suitable for flows with moving shocks? (b) What are the model hyperparameters? How do they affect accuracy? (c) How many snapshots are required to achieve predictive accuracy of* $< 5\%$*?*

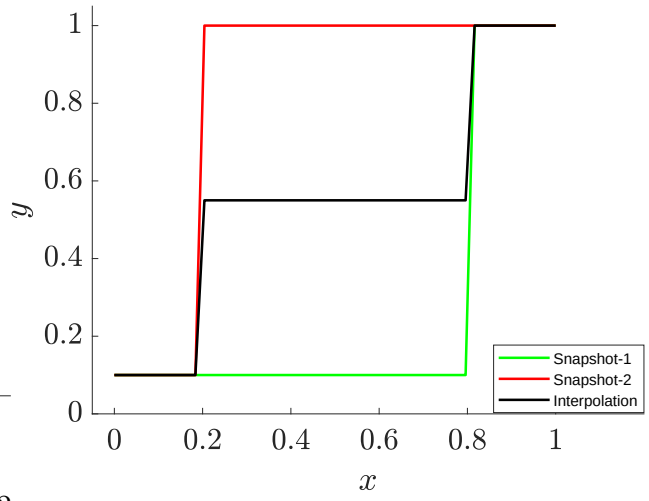
Flows with moving shocks

The POD method used for model reduction has an inherent limitation due to its linear embedding. Since it assumes that the state variable is always in the subspace formed by a finite set of orthonormal basis vectors which are in turn extracted from a finite set of arbitrary flow snapshots, unless a *perfect* basis set is constructed, the state might not always be predicted with acceptable accuracy. While there are approaches such as *dense space-filling* designs and *sequential adaptive designs* that can provide a practically useful set of snapshots to construct the POD basis vectors, when the flow contains non-linearities in the form of discontinuities (such as shocks), it poses additional challenges. This is primarily because with moving shocks, the linear assumption in the method tends to diffuse the shock at a given location, rather than accurately predicting the discontinuity and its spatial location.

To see, this consider a simple example used to demonstrate this behavior. In Figure 4.21a, 2 snapshots that represent a discontinuity are shown (by the red and green lines). These can be considered analogous to the transonic flows which the shock locations varying



(a) 2 flow snapshots with discontinuities at $x = 0.2$ and $x = 0.8$



(b) Prediction (via interpolation) at $x = 0.5$

Figure 4.21: Demonstration of POD applied to solution with discontinuities. 2 snapshots are used (red and green lines) is shown to the left and the 2-mode interpolation to predict an intermediate solution at $x = 0.5$ is shown to the right. Notice that the interpolated solution linearly combines both snapshots to result in a stair-step like prediction. In other words, the POD-based method tends to *diffuse* the discontinuity

due to parameter sensitivity. The linearly interpolated state [1] (black line in Figure 4.21b) can be seen to combine the effects of both the snapshots, showing a *stair-step* like variation in the state. In the limit of infinitely many snapshots, the approach would be able to capture the discontinuity accurately, but in all other cases, such a behavior is inevitable.

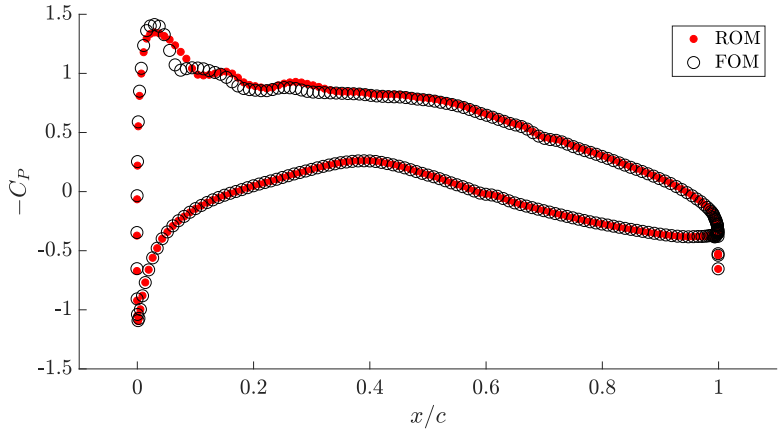
In such cases, a linear method such as POD might have to be replaced with a non-linear method that is more appropriate. Similar issue has been reported by other works; who address it through a domain-decomposition method in which they solve the high-fidelity model (FOM) at a localized domain where highly non-linear phenomena are expected while solving the ROM in the rest of the domain. See [71, 72, 73, 133, 134] for further details. Such an approach is deliberately avoided in this work in order to keep the overall method non-intrusive and widely applicable. Additionally, the present method does not explicitly specify the boundary conditions in the ROM and rather lump them along with the RHS

[1] The POD coefficients (coordinates) are interpolated between the 2 snapshots to predict the state as a combination of the POD modes

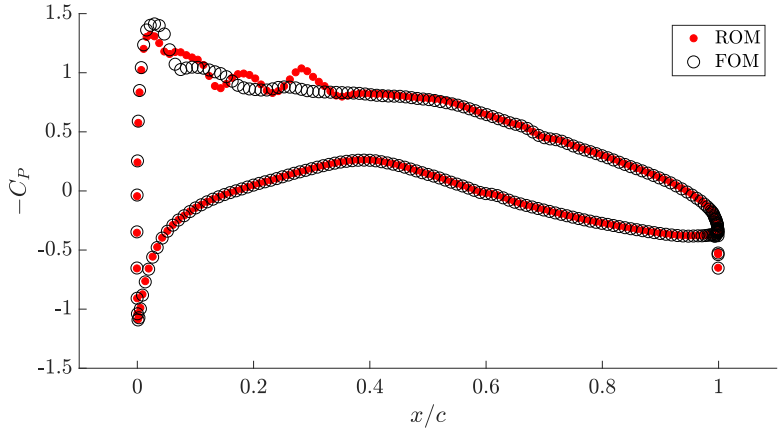
(for the same reason as to keep the approach non-intrusive) which is expected to incur error in flow conditions that are expected to be highly sensitive to boundary conditions. However, despite that, the present method is able to predict the shock location and strength within approximately 5% of the FOM. Due to the large sensitivity of the drag coefficient to discrepancy in the pressure distributions and the error it incurs, the applicability of the present method to PDE-constrained optimization problems involving moving shocks could be problematic. Some special care should be taken to improve accuracy of the method in such flow conditions, before applying them to many-query contexts.

Effect of the interpolation step

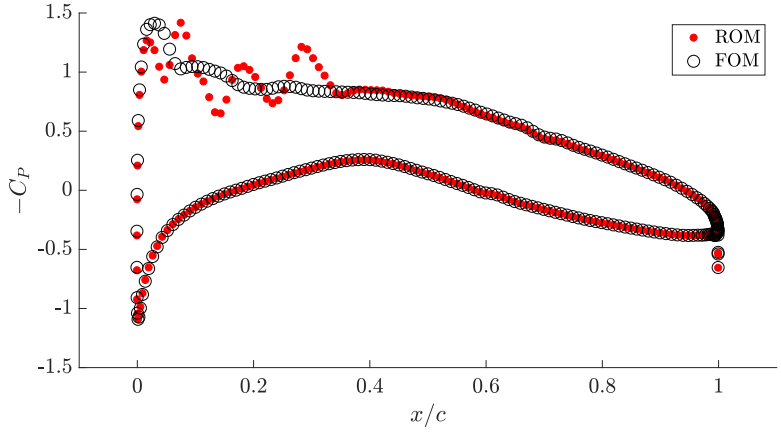
The key to success of the present method is an accurate interpolation of the system matrices at new parameter values. With only flow parameters present in the system, the linear operator matrix is invariant across different snapshots (since the mesh is fixed) and hence interpolation is done only for the RHS of the ROM ($\tilde{\mathbf{f}}$). As mentioned in Chapter 2 interpolation is performed using polynomials in the Lagrange form, where the order of polynomial that is appropriate is problem dependent. Additionally, it is also limited by the number of snapshots available (see Chapter 2). However, within the permissible order of polynomials for interpolation, the correct value again is problem dependent. For the RAE2822 test case for instance, the effect of interpolation on the pressure coefficient prediction is shown in Figure 4.22. For this specific test case, 1^{st} order polynomials lead to a better prediction than 2^{nd} or 3^{rd}. However, for the NACA0012 case, the best prediction is observed with a 2^{nd} order interpolant. Therefore, while the interpolatory property of the present approach has the advantage of being non-intrusive, it has the limitation that the best interpolant for a given problem might not be known apriori and is an open question at the moment. However, within the experience of the work done in this thesis, one among linear, quadratic and cubic interpolation always lead to satisfactory error in prediction.



(a) ROM interpolation with 1^{st} order polynomial



(b) ROM interpolation with 2^{nd} order polynomial



(c) ROM interpolation with 3^{rd} order polynomial

Figure 4.22: Effect of the ROM interpolation on its predictive capability

Influence of the snapshot size

The number of snapshots used for ROM training was 40 in the NACA0012 case which seemed sufficient in order to predict the non-linear flow field with error in $\mathcal{O}(1\%)$. How-

ever, with the inclusion of highly non-linear phenomenon in the flow-field such as shocks, it was necessary to ensure adequate resolution of the state space is used in the training phase. For the RAE2822 test case, 80 snapshots (twice as many as the subsonic NACA test case) in addition to a more localized parameter space in the high-transonic mach number regime was necessary in order to predict the shock location within $2-5\%$ chord lengths. Additionally, it was observed that larger snapshot sizes were mainly driven by the error incurred in C_d. In Figure 4.23, the *average* error in prediction for the 4 snapshots sizes tested for the transonic RAE2822 case are compared for C_d, C_l and C_P. It can be seen that with a sparse snapshot set, the error in C_d is very high which improves considerably with more snapshots. However, with C_l and C_P, (while still slightly improving with more snapshots) the error is more or less uniform across all the 4 test cases. This is again due to greater sensitivity in the C_d predictions as explained previously. Overall, under transonic conditions, the method still predicts the field variables, C_P and C_l with accuracy $\sim 10\%$ while similar accuracy in C_d requires more snapshots.

Despite some investigations on the training data size, the number of snapshots used in this work are arbitrary in the sense that they were not determined after rigorous experimentation. The author expects that the appropriate number of snapshots necessary is again, problem dependent and hence an adaptive approach to model development using criterion-based designs [135] could ensure a reasonable snapshot size is used for each problem. Finally, in this thesis a lower bound for the snapshot size based on the requirements for the interpolation is provided as $\varrho = \binom{n+m}{m}$ where n is the order of the polynomial interpolant and m is the number of design parameters. While such a lower bound is certainly not universal, it served as a means to arrive at a good initial guess for the snapshot size, which may further be improved upon iteratively.

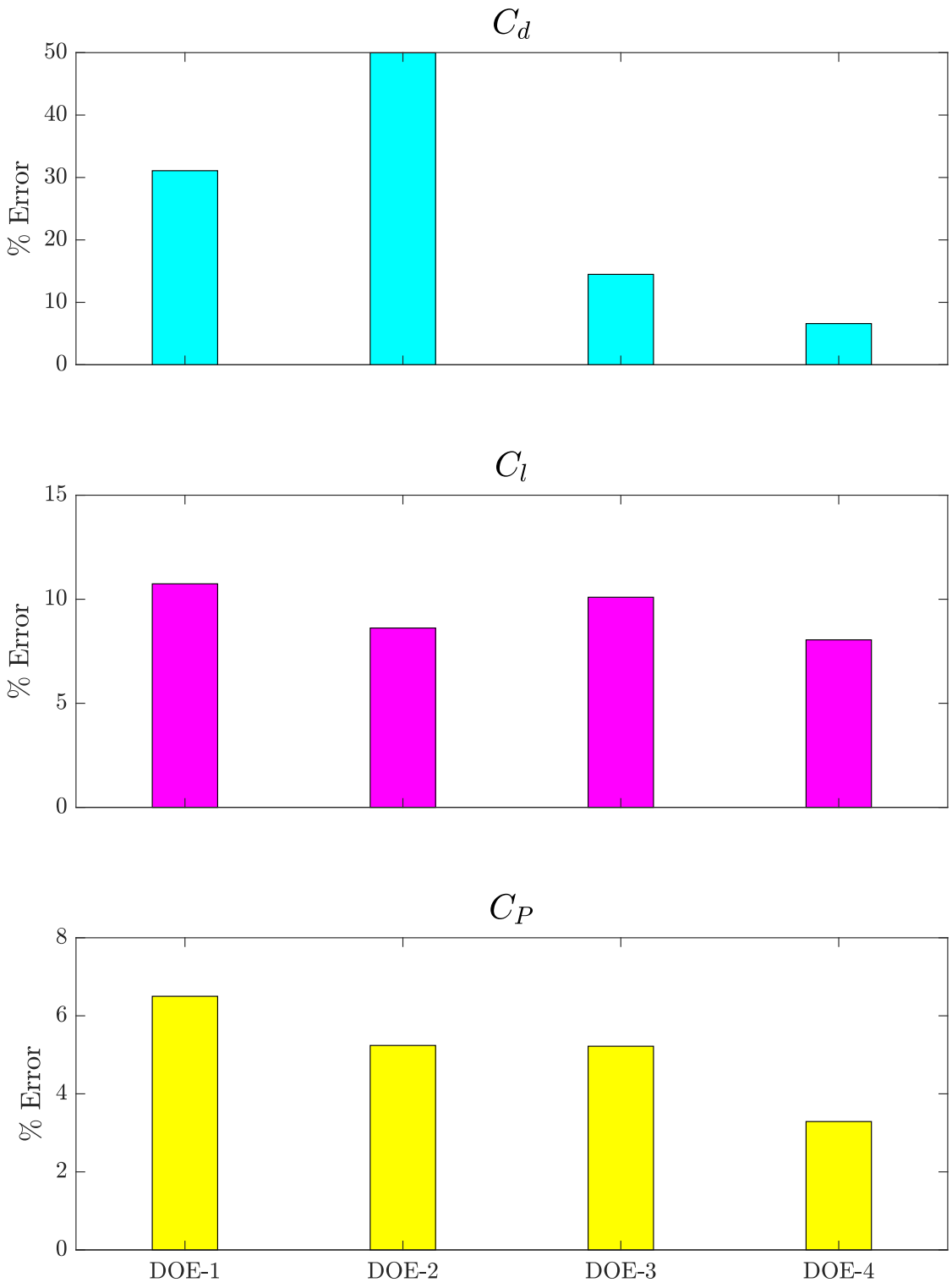
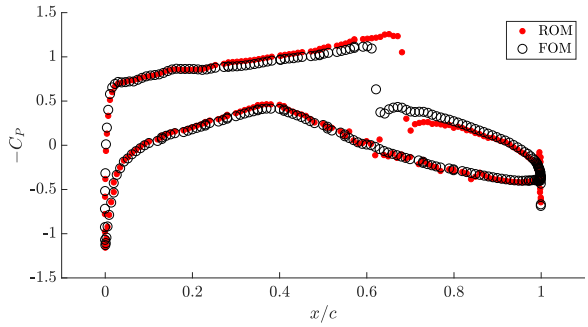


Figure 4.23: Effect of snapshot size on model performance in the transonic regime

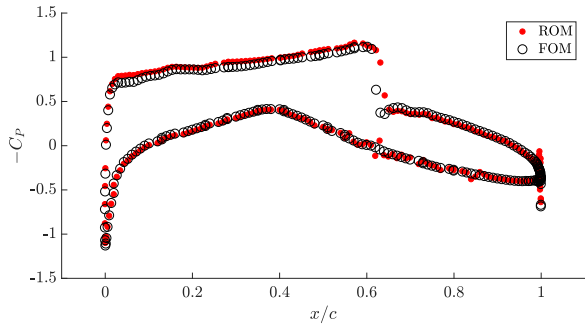
Influence of the choice of initial guess

A challenge in the proposed ROM methodology is the well-behaved convergence of the solver. The non-linear constrained optimization problem (Eq. 4.7 is solved via an iterative procedure that is sensitive to the user-provided initial guess. Specifically, in the high-transonic regime, it was observed that the state-space is sensitive to the parameters - which is essentially manifested as different shock locations and shock strengths depending on the the value of $[\mathbb{M},\ \alpha]$. Further, the interpolation of the system matrices ($\tilde{\mathbf{f}}$ in this case) is certainly expected to incur error as the true nature of their space is unknown in the case of black-box models. Additionally, the SQP method used to solve the optimization problem for the ROM is guaranteed to only provide a local optimum, in the neighborhood of the initial guess. To demonstrate this, see Figure 4.24 where the flow at $\mathbb{M} = 0.74$ and $\alpha = 0.95\ deg.$ is predicted via the ROM at 2 different initial guess. In both cases, the convergence tolerance on the scaled objective function and constraints were set to $1e-6$. While both solutions converged, the predicted results are not exactly the same.

In order to remedy this issue in the present approach, the initial guess solution is chosen as one of the training snapshots. Further, the snapshot whose parameter minimizes the *standardized euclidean distance* to the parameter for which the ROM is being solved for is chosen. When the parameters of the problem could have different different orders of magnitude, a regular euclidean distance minimization could bias the search for the initial guess towards the larger parameter. Further, each parameter could have a different influence on the flow - for instance at $\mathbb{M} = 0.7$, a change in α from $0-1$ might not change from a shock free flow to one with a strong shock. On the other hand, at $\alpha = 3$, a change from $\mathbb{M} = 0.7$ to $\mathbb{M} = 0.8$ could introduce a shock. Numerically speaking, the change in mach number was 0.1 which is much smaller than a change in α of 1, however, the influence on the physics could be drastic. Therefore the right choice of initial guess is paramount. Within the cases tested in this study, the standardized euclidean which scales the distance between points by the standard deviation of each coordinate gave best results than a regular



(a) Relative error on C_P is $\approx 25\%$



(b) Relative error on C_P is $\approx 3.5\%$

Figure 4.24: Effect of initial guess on the ROM prediction. Flow conditions: $\mathbb{M} = 0.74$, $\alpha = 0.95\ deg$.

euclidean. For instance, see Figure 4.25; the red point is the nearest to the query point (black) in the euclidean sense. However, if the data are corrected for their spread, the green point (standardized euclidean) is the nearest to the query point. If the variable $\mathbf{x}_1$ is assumed to be the mach number, then the green and black points are more likely to be similar in physics compared to the red and black points.

Extrapolative capabilities

The fundamental ansatz of the POD-based ROM methodology is that it searches for the solution within the subspace spanned by the snapshots themselves. Therefore in principle, they are not expected to predict the solution outside of the parameter range used for the snapshots. However, it is always of interest to evaluate how good (or bad) the extrapolative capabilities of the ROM are; a specific situation of interest is in the case of multi-disciplinary design optimization where the process of iterating between disciplines

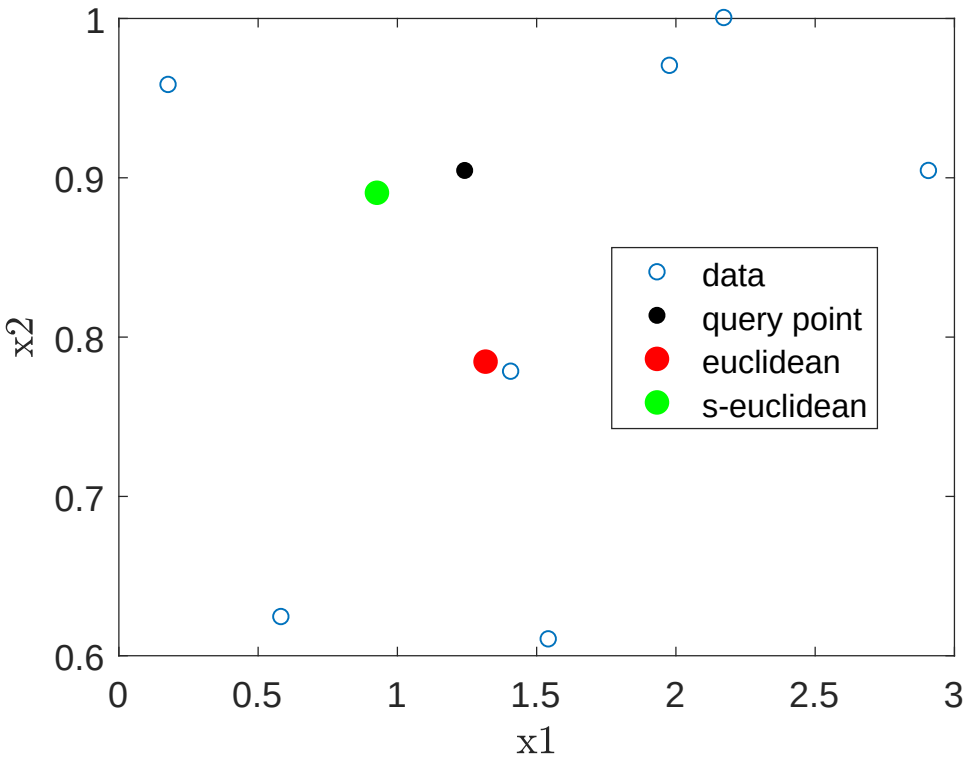


Figure 4.25: Impact of *similarity metric* used to find initial guess for ROM

could occasionally throw the system out of an apriori anticipated parameter space. In such cases, the robustness of the ROM with respect to parameter choices outside of the snapshots is critical.

4.2 Variation in shape parameters

Having successfully validated the parametric reduced order modeling methodology with flow parameters, the method is now tested with geometry parameters being varied. Introducing shape parameters makes the problem interesting in several ways; firstly the linear operator matrix becomes parametric and hence a suitable matrix interpolation method is necessary, secondly shape parameters introduce a high-dimensional design space on which the methodology can be tested and finally it prepares the methodology to be applied towards aerodynamic shape optimization problems. The airfoil shape is parametrized using a suitable set of parameters, while keeping the flow parameters fixed.

Research Question 6. *How does the proposed approach compare against existing approaches?*

Parametrizng the airfoil shape is non-trivial specifically due to regions such as the rounded leading edge (that could lead to infinite slope) and the sharp trailing edge (which could lead to a discontinuity). A parametrization that can accurately approximate any airfoil shape while also keeping the number of parameters minimal and exhibiting smooth variation to parameter changes is desired. The *Class Shape Transformation (CST)* originally proposed by Kulfan [136, 137] is adapted in this work and is briefly explained in what follows [2].

4.2.1 Class Shape Transformation (CST)

The CST model of parametrization defines a *class* function C and a *shape* function S and the curve being parameterized is specified as their product. The main idea is that the class function serves to define a general class of geometry such as airfoils, missiles or sears-haack body, while the shape function serves to define the unique shape within a particular

[2] Alternate parametrizations that lead to smooth shape variations due to parameter perturbations are also equally applicable

class of shapes (such as a NACA0012 vs RAE2822 airfoil). The class function, $C(\psi)$ is more generally defined as

$$C_{n_1}^{n_2}(\psi) := \psi^{n_1}(1-\psi)^{n_2} \tag{4.10}$$

where the variable ψ represents the non-dimensional chord-wise distance. n_1 and n_2 define the specific class; for instance $n_1 = 0.5,\ n_2 = 1$ and hence $\sqrt{\psi}(1-\psi)$ defines airfoils with rounded leading edge and a sharp trailing edge [136]. More specifically, the $\sqrt{\psi}$ term controls the leading edge shape and the $1-\psi$ term controls the trailing edge shape. The unique shape of an airfoil is driven by the shape function, specified as follows

$$S(\psi) = \sum_{i=0}^{n} A_i \psi^i \tag{4.11}$$

It is particularly useful to define a *unit shape function*, i.e. $S(\psi) = 1$ such that the individual coefficients A_i [3] can be obtained as generic constants. For instance for $n = 1$ the simplest decomposition one could get for the shape function is $S(\psi) = S_0(\psi) + S_1(\psi)$ where $S_1(\psi) = \psi$ and $S_2(\psi) = 1-\psi$ where the coefficients $A_0 = 1$ and $A_1 = 1$. Similarly, for the general nth order shape function, the decomposition of the unit shape function can be done using *Bernstein* polynomials

$$S(\psi) = \sum_{i=0}^{n} K_{i,n} \psi^i (1-\psi)^{n-i} \tag{4.12}$$

where the coefficients are the *binomial* coefficients given by

$$K_{i,n} = \binom{n}{i} = \frac{n!}{i!\,(n-i)!}$$

[3] A_i's are denoted by $K_{i,n}$ for unit shape functions

The final shape of the airfoil shape is then given by

$$\mathbf{y}(\psi) = C(\psi)S(\psi) \tag{4.13}$$

The unit shape functions and the corresponding airfoil geometries are illustrated in the Figure 4.26. It can be seen that such a parametrization results in each component shape function peak being equally distributed between the leading and trailing edges leading to the same effect in the component airfoils. It is now a matter of scaling up or down, the binomial coefficients of the Bernstein polynomials in order to approximate the unique airfoil shape of interest.

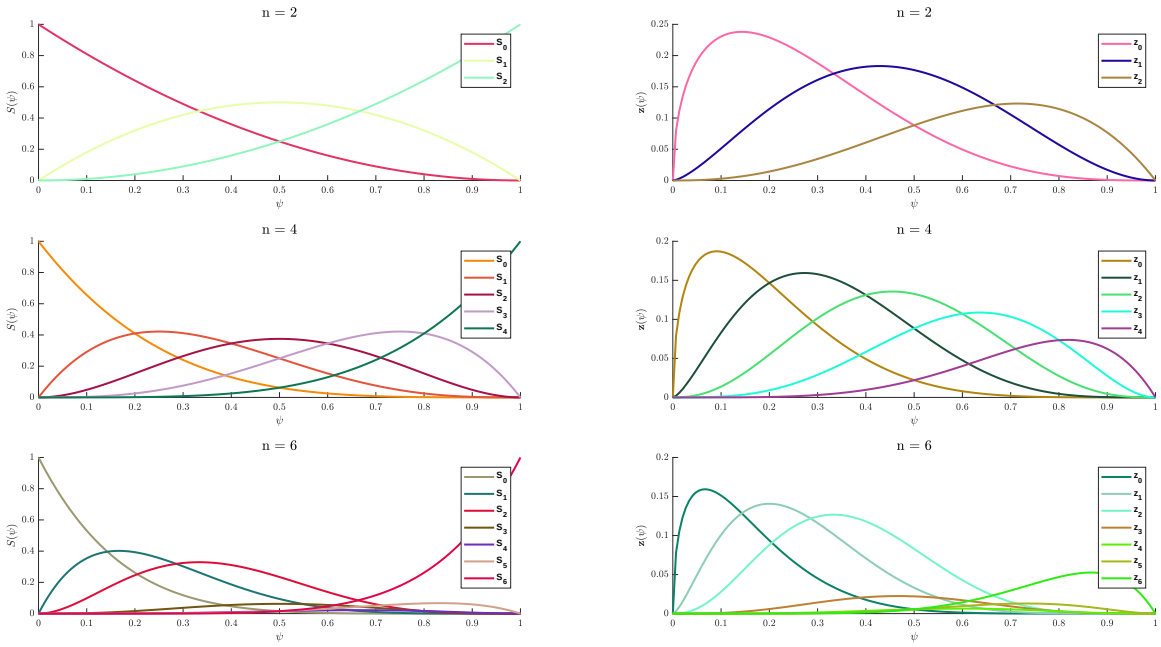


Figure 4.26: Examples of the shape function decomposition (into Bernstein polynomials) for various values of the order n (*left*) and the resulting component airfoils (*right*). The coefficients A_i correspond to unit shape function, which can be scaled up/down to obtain a specific airfoil shape.

The coefficients A_i represent the actual parameters of the shape, given n the order of the Bernstein polynomials. An nth order CST parametrization has $n + 1$ parameters. If separate parametrizations are sought for the upper and lower surfaces of the airfoil, then the CST parametrization leads to $2(n + 1)$ parameters to specify the whole shape of the

airfoil, where the n needs to be determined for a specific geometry under consideration. However, typically $n = 3 - 5$ are observed to be adequate to parametrize the airfoil shapes considered in this work. One way to determine n and the associated polynomial coefficients is to find the values that minimizes certain error between the true shape of the airfoil and the resulting approximation via CST. In this work, the parameters for a given airfoil shape are determined by solving the following minimization problem

$$\underbrace{\text{minimize}}_{A_j,\, n} \quad \left\| \mathbf{y}(\tilde{\psi}) - C(\tilde{\psi}) S(\tilde{\psi},\ A_j) \right\|_2^2 \tag{4.14}$$

where $\tilde{\psi} \in \mathbb{R}^{n+1}$ are $n+1$ equally spaced points sampled from ψ spanning $[0, 1]$ [4]. This way, the smallest possible n and their corresponding Bernstein coefficients are determined. For the RAE2822 airfoil shape, the following parameterization was obtained ($n = 3$):

$$A = \begin{bmatrix} 0.1268 & 0.4670 & 0.5834 & 0.2103 \\ -0.1268 & -0.5425 & -0.5096 & 0.0581 \end{bmatrix}$$

where the first and second rows represent the parameterization of the upper & lower surfaces of the airfoil; the comparison of the CST curve and the actual RAE2822 shape is shown in Figure 4.27. It can be seen that the CST parametrization gives an *adequate* approximation to the true curve with only 8 parameters. The coefficients may now be perturbed to modify the baseline airfoil shape.

Similarly, the NACA0012 airfoil shape is approximated via CST and is shown in Figure 4.28. In this case, due to the lack of camber, the CST gives very good approximation with $n = 2$. Additionally, due to the symmetry of the airfoil about the chord, the parameters (given below) are equal in magnitude and opposite in sign. However, all the $2(n + 1)$ degrees of freedom are considered in this work for the NACA0012 airfoil.

[4] Note that picking equally spaced points works well for the current airfoil geometries under consideration in this work and may not work for any arbitrary geometry. More generically, a least-squares fit considering all the points might be more suitable

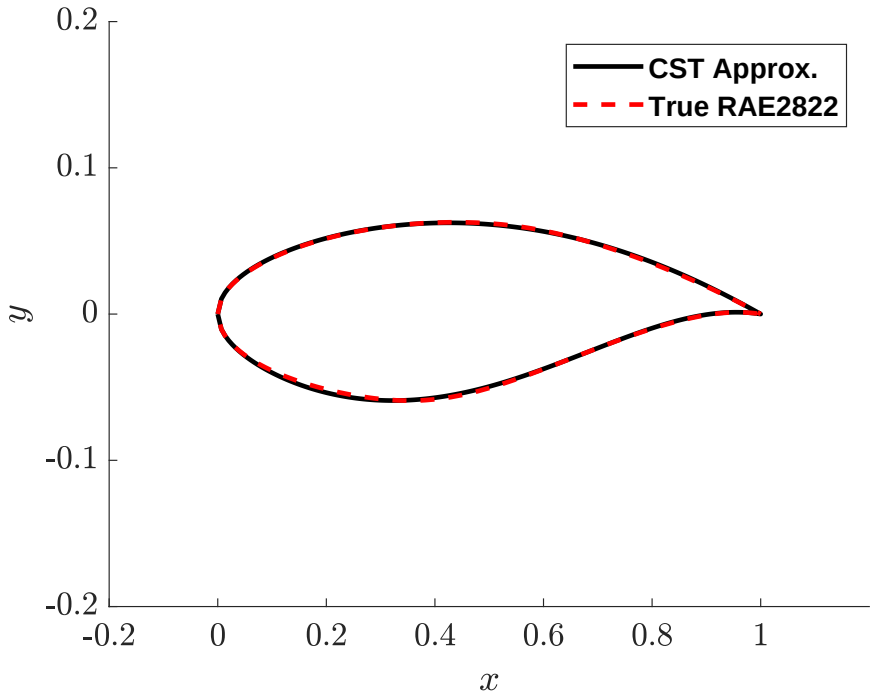


Figure 4.27: Comparison of the CST approximation to RAE2822 against the true curve

$$A = \begin{bmatrix} 0.1689 & 0.2699 & 0.1387 \\ -0.1689 & -0.2699 & -0.1387 \end{bmatrix}$$

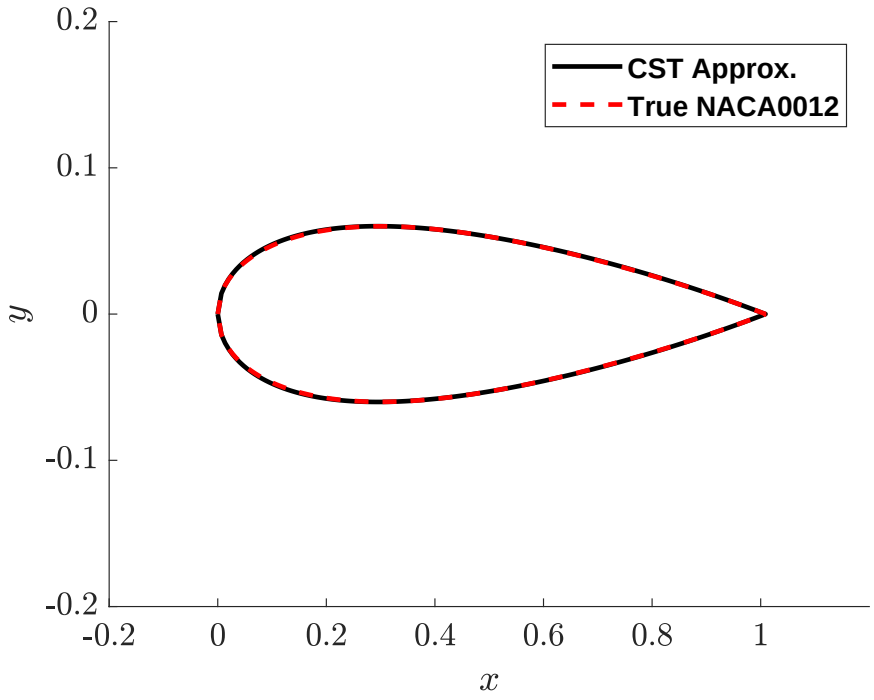


Figure 4.28: Comparison of the CST approximation to NACA0012 against the true curve

4.2.2 NACA0012

The CST coefficients representing the NACA0012 baseline is perturbed $\pm$ 30% to generate new airfoil shapes, a sample of which is shown in Figure 4.29. A total of 170 such points were generated, 160 of which was used in model building while the remaining was used to validate the model. A couple of flow snapshots are shown in Figure 4.30. The system matrix $\tilde{\mathbf{B}}$ is interpolated in the tangent space to the manifold of symmetric positive definite matrices, as explained in Chapter 2, while the RHS $\tilde{\mathbf{f}}$ is interpolated in the Euclidean space. In both cases a multi-variate polynomial in the Lagrange form is used for interpolation.

The comparison of the ROM predicted pressure coefficient on the airfoil surface against the FOM solution is shown in Figure 4.32. An accurate prediction of the pressure coefficient is observed. Across all the 10 validation cases, the maximum and average errors in C_P are $\approx 5\%$ and $\approx 2\%$ respectively, which is sufficiently accurate given the computational speedup of $\approx$ 100x in solving the ROM. Further, the lift coefficient is also computed from the ROM predictions and match the FOM predictions with similar accuracy to that of C_P,

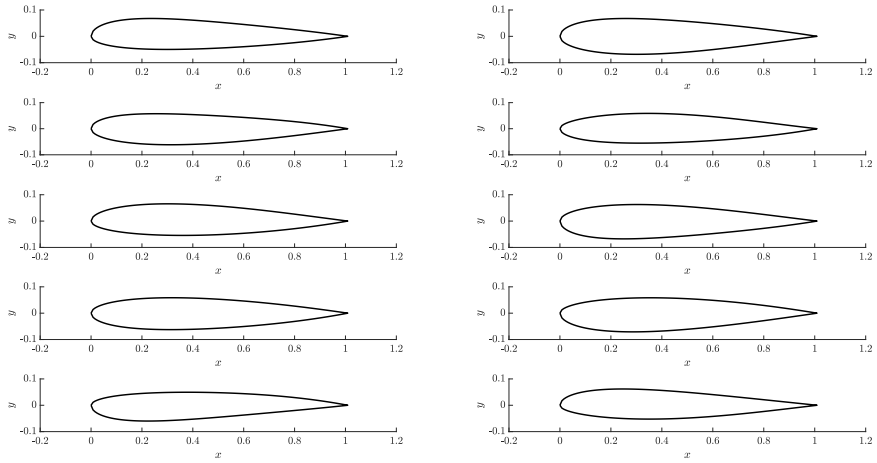
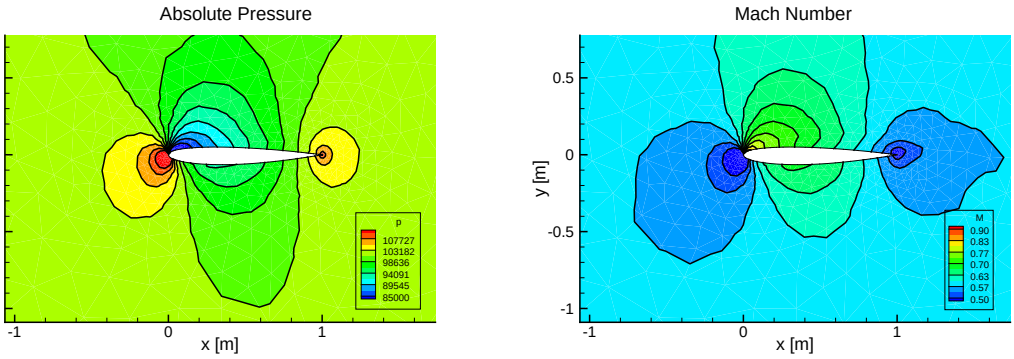


Figure 4.29: Family of airfoils generated by perturbing (by $\pm 30\%$) the CST coefficients of the NACA0012 baseline

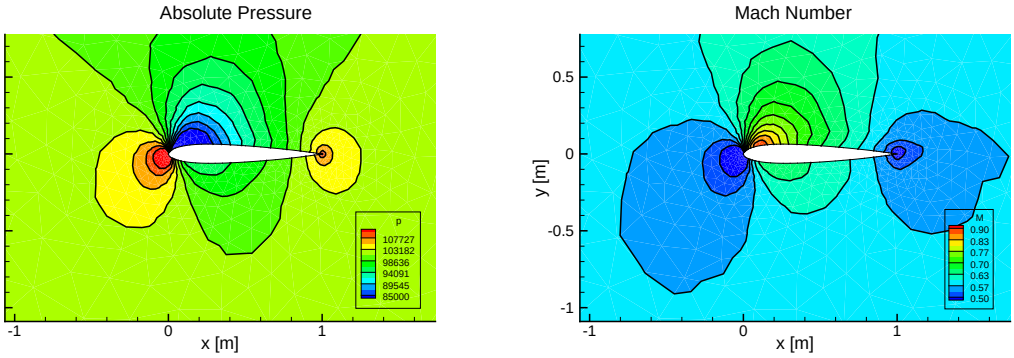
Table 4.10: Free-stream conditions for the RAE2822 test case

P_∞	101325 Pa
ρ_∞	1.225 kg/m^3
a_∞	340.296 m/s
μ_∞	1.785E-5 $Pa-s$
$\mathbb{M}$	0.6
α	2.0 $deg.$

and are provided in Table 4.11. The overlaid pressure and mach number contours are shown in Figure 4.31 which re-iterate the low prediction errors observed from the validation cases.



(a) Flow snapshot corresponding to CST coefficients $[0.1262,\ 0.2296,\ 0.1352,\ -0.1482,\ -0.3310,\ -0.1724]$

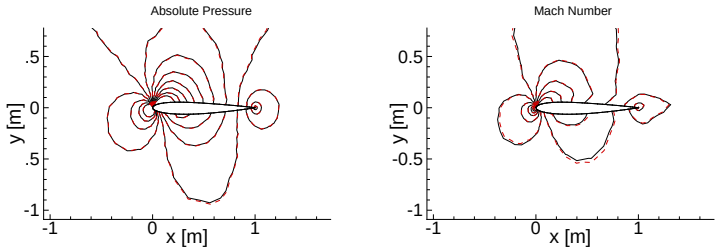


(b) Flow snapshot corresponding to CST coefficients $[0.2167,\ 0.2061,\ 0.1617,\ -0.2015,\ -0.2070,\ -0.1692]$

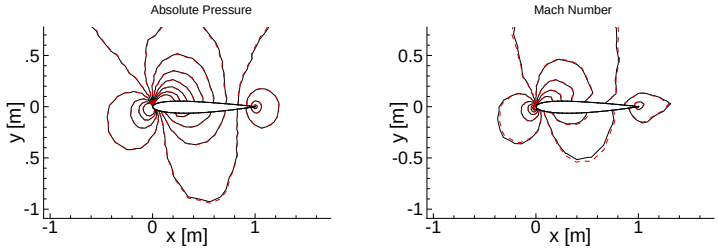
Figure 4.30: Sample flow snapshots (corresponding to shape parameters) for the NACA0012 test case. Each row represents a different airfoil shape.

Table 4.11: Comparison of C_P, C_l and C_d between ROM & FOM for the NACA0012 test case

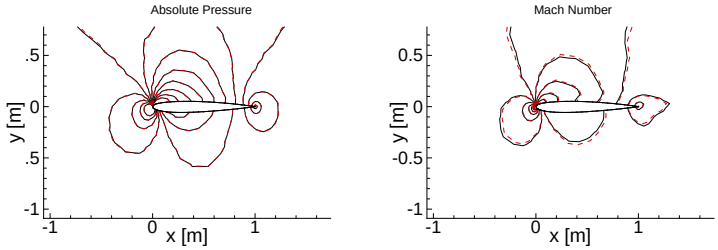
Case	C_P Error %	C_l (ROM)	C_l (FOM)	Error %
1	1.29	0.1889	0.1912	1.20
2	0.74	0.2018	0.2070	2.50
3	0.80	0.2932	0.2943	0.37
4	1.86	0.2795	0.2865	2.44
5	1.36	0.3550	0.3621	1.96
6	0.62	0.3691	0.3664	0.73
7	0.46	0.3298	0.3272	0.79
8	2.79	0.3229	0.3312	2.50
9	0.76	0.3109	0.3137	0.89
10	5.43	0.2710	0.3065	11.58



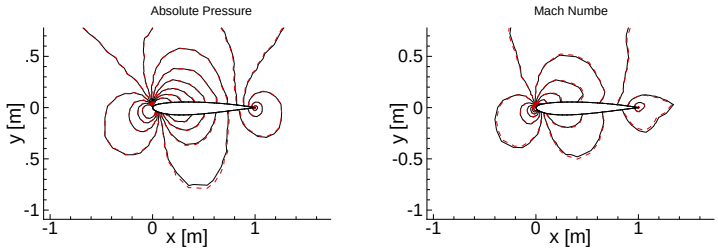
(a) Val. Case-1



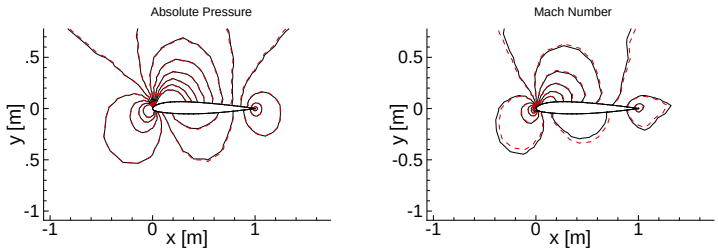
(b) Val. Case-2



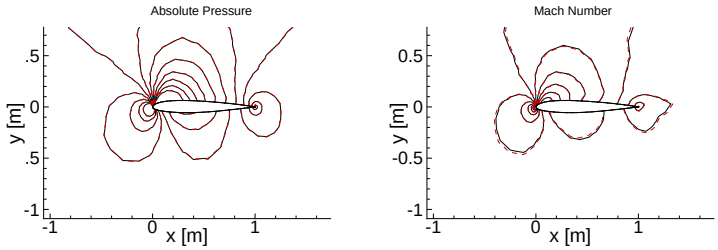
(c) Val. Case-3



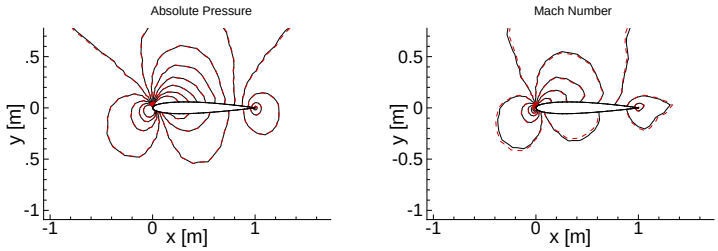
(d) Val. Case-4



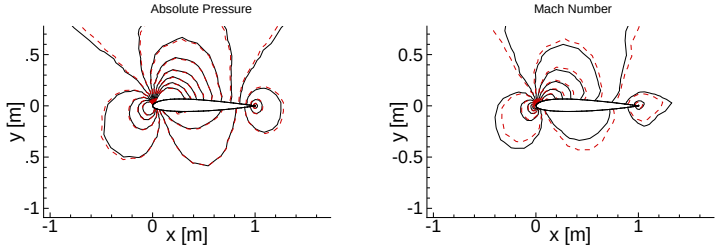
(e) Val. Case-5



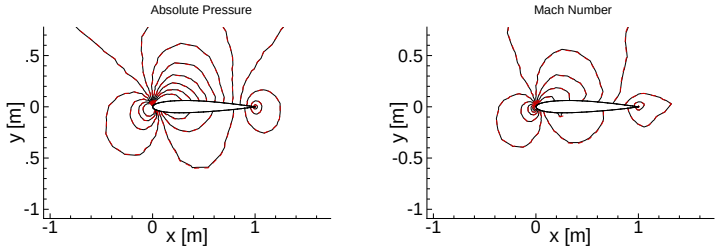
(f) Val. Case-6



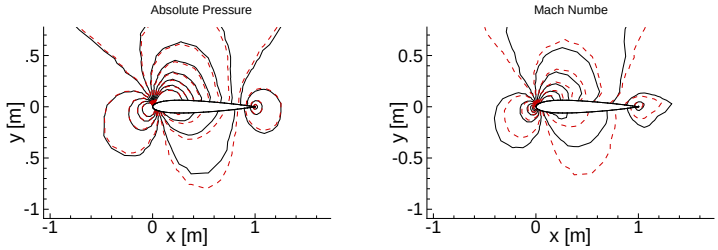
(g) Val. Case-7



(h) Val. Case-8

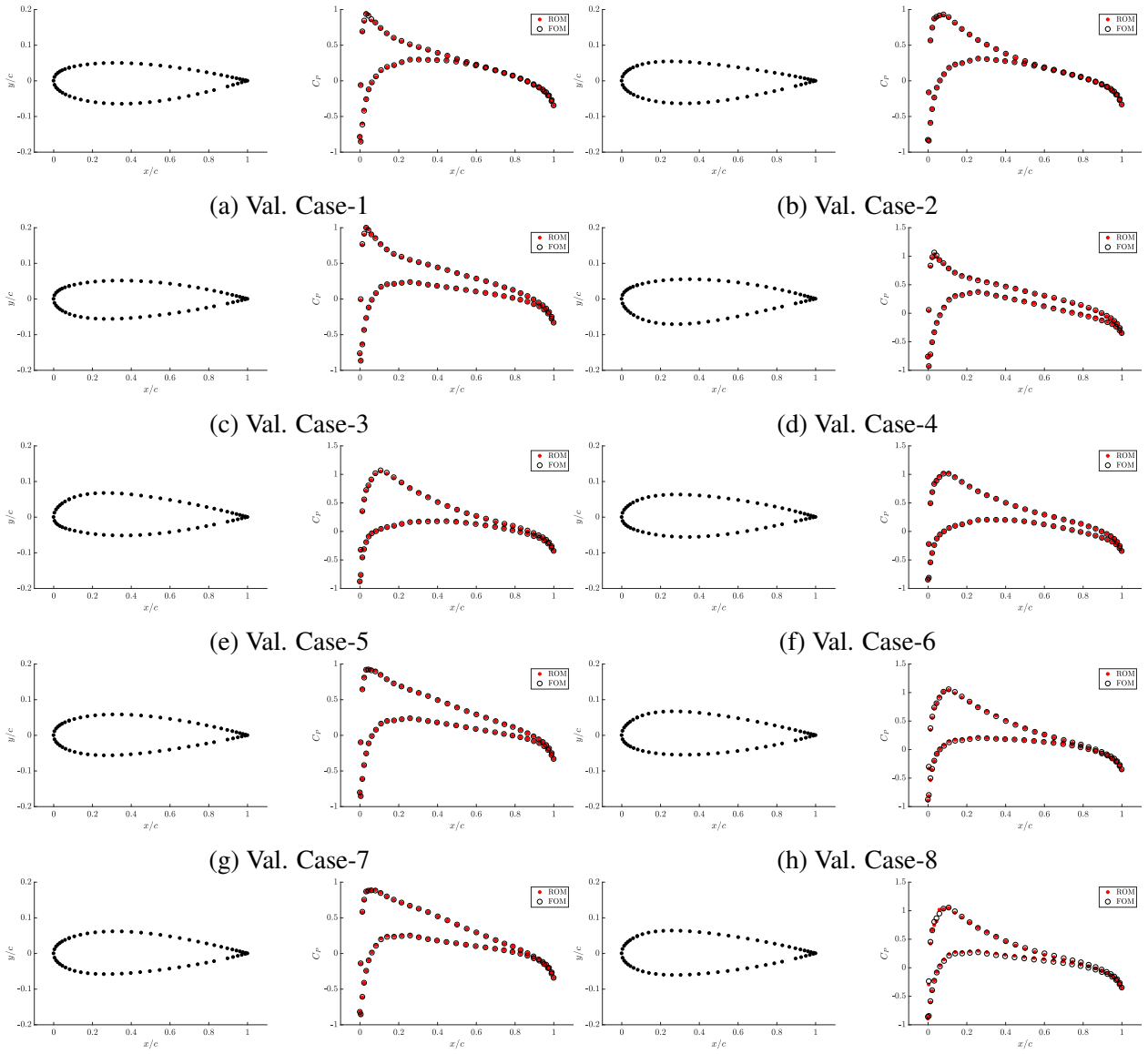


(i) Val. Case-9



(j) Val. Case-10

Figure 4.31: Comparison of Mach number and absolute pressure between ROM & FOM with respect to shape parameters for the NACA0012 test case



(a) Val. Case-1

(b) Val. Case-2

(c) Val. Case-3

(d) Val. Case-4

(e) Val. Case-5

(f) Val. Case-6

(g) Val. Case-7

(h) Val. Case-8

(i) Val. Case-9

(j) Val. Case-10

Figure 4.32: C_P comparison between ROM & FOM for various shape parameters for the NACA0012 test case

Table 4.12: Free-stream conditions for the RAE2822 test case

P_∞	28,745 Pa
ρ_∞	0.44 kg/m^3
a_∞	301.86 m/s
μ_∞	1.49E-5 $Pa-s$
$\mathbb{M}$	0.734
α	2.79 $deg.$

4.2.3 RAE2822

The CST coefficients representing the RAE2822 baseline are perturbed $\pm$ 30 % to generate new airfoil shapes, a sample of which is shown in Figure 4.33. A total of 170 such points were generated, 160 of which were used in model building while the remaining 10 were used to validate the model. The interpolation of the ROM for parameter changes are same as that for the NACA0012 case. However, the freestream mach number for this case is set as $\mathbb{M} = 0.734$ which leads to a shock towards the leading edge of the airfoil whose strength and location are affected by perturbing the shape CST coefficients. Sample flow snapshots are shown in FIgure 4.34. The rest of the freestream conditions are summarized in Table 4.12

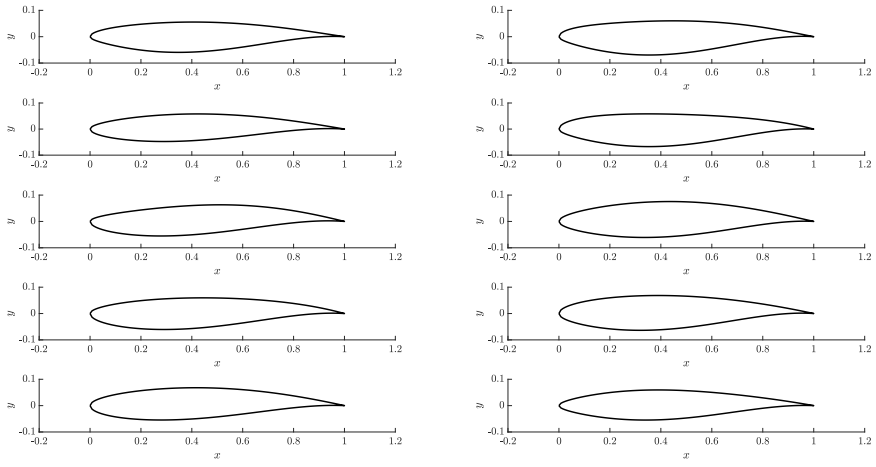
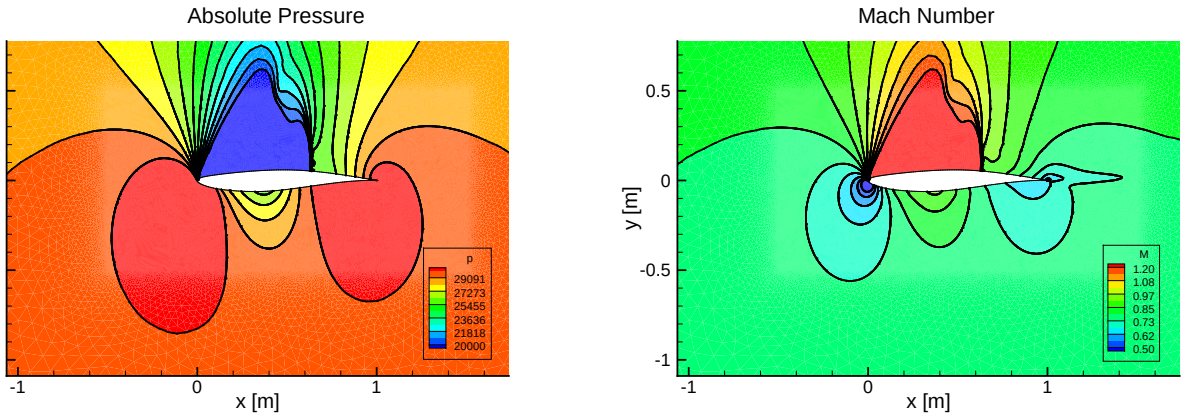
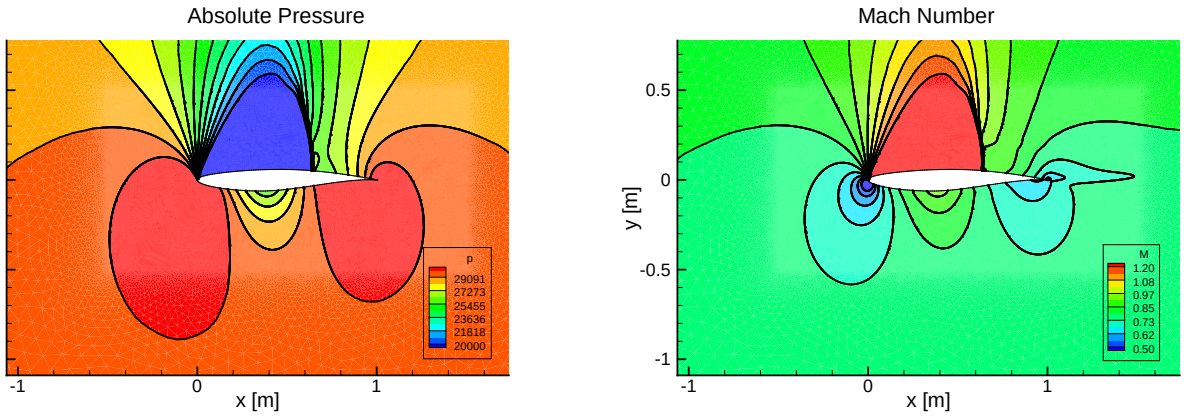


Figure 4.33: Family of airfoils generated by perturbing (by $\pm 30\%$) the CST coefficients of the RAE2822 baseline

We begin by comparing the ROM predictions against FOM solutions in terms of flow



(a) Flow snapshot corresponding to CST coefficients $[0.0897,\ 0.3650,\ 0.6776,\ 0.1524,\ -0.1522,\ -0.5415,\ -0.4490,\ 0.0615]$



(b) Flow snapshot corresponding to CST coefficients $[0.1086,\ 0.4015,\ 0.5844,\ 0.1935,\ -0.1342,\ -0.5492,\ -0.5395,\ 0.0638]$

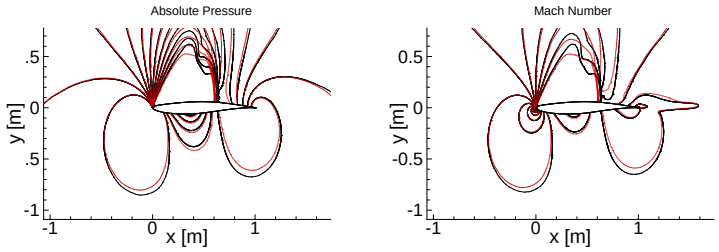
Figure 4.34: Sample flow snapshots for the RAE2822 test case. Each row represents a different airfoil shape.

contours. In Figure 4.35, the pressure and mach number contours predicted by the ROM are compared against the true FOM solution, as overlaid contour plots. The ability of the ROM to accurately capture the flow field is emphasized from these plots. The comparison of the ROM and FOM solutions, with respect to pressure coefficient contours is shown in Figure 4.36. These plots corroborate the observations from the overlaid contours. Further these plots visualize the shock strength in terms of the jump in the pressure coefficient. Overall, the ROM captures both the shock location (within 5% chord lengths) and the shock strength accurately.

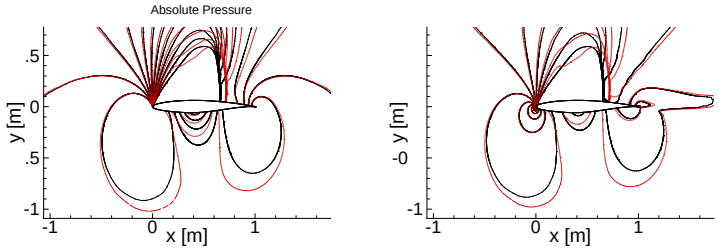
Table 4.13: Comparison of C_l and C_d predicted by ROM against the FOM

Case	C_P Error %	C_d (ROM)	C_d (FOM)	Error %	C_l (ROM)	C_l (FOM)	Error %
1	7.68	0.0161	0.0174	7.47	0.9174	0.9608	4.52
2	12.93	0.0336	0.0302	11.26	1.0669	0.9825	8.59
3	2.84	0.0257	0.0262	1.91	1.1140	1.1446	2.67
4	8.14	0.0321	0.0264	21.59	1.1004	1.0901	0.95
5	13.77	0.0194	0.0224	13.39	0.8965	1.0345	13.34
6	12.78	0.0298	0.0484	38.43	1.0288	1.0082	2.04
7	5.91	0.0306	0.0279	9.68	0.9403	0.9286	1.26
8	4.99	0.0245	0.0245	0	0.9124	0.8914	2.36
9	5.63	0.0292	0.0315	7.30	0.9507	0.9779	2.78
10	9.96	0.0326	0.0217	50.23	0.9750	0.9793	0.44

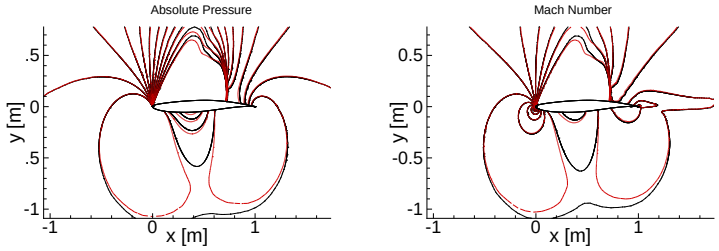
However, the more pertinent question within the context of this thesis really is about the ROM's capability to predict the drag coefficient with sufficient accuracy as the FOM - this is more important from the point of view of design optimization. The Table 4.13 summarizes this comparison. The C_P is predicted with an error of 8.5% on average. The ROM predicts the C_d within $\approx$ 16% error on average, while the C_l is captured within 4 % on average. The higher error in drag is mainly contributed by the last validation case which incurs about 50% error.



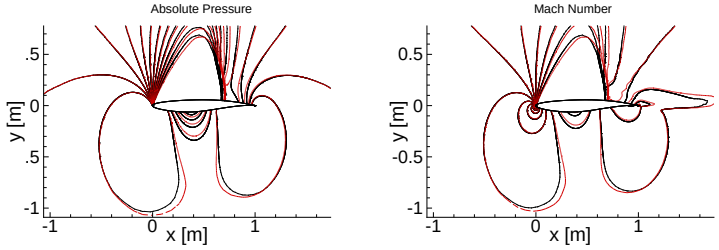
(a) Val. Case-1



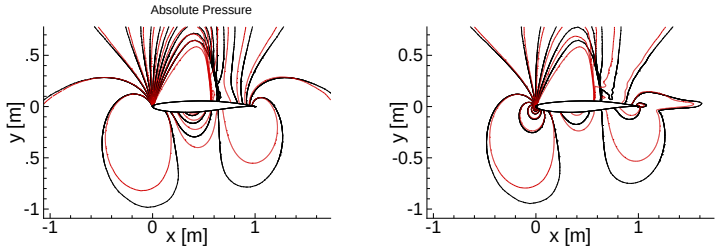
(b) Val. Case-2



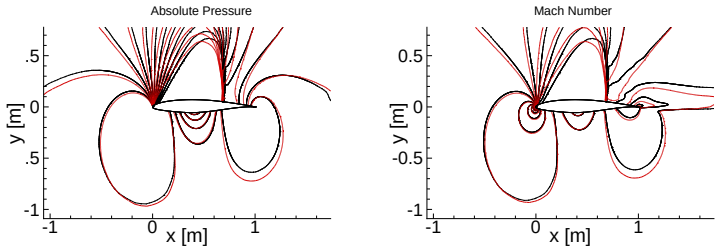
(c) Val. Case-3



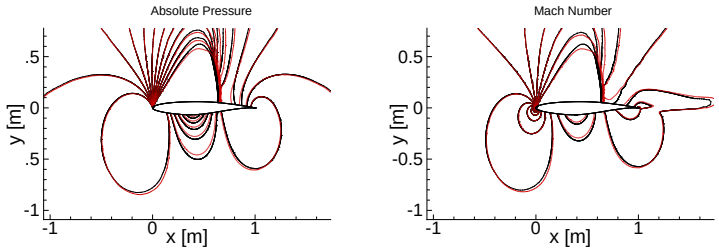
(d) Val. Case-4



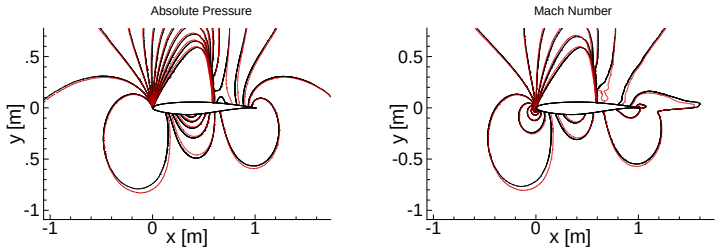
(e) Val. Case-5



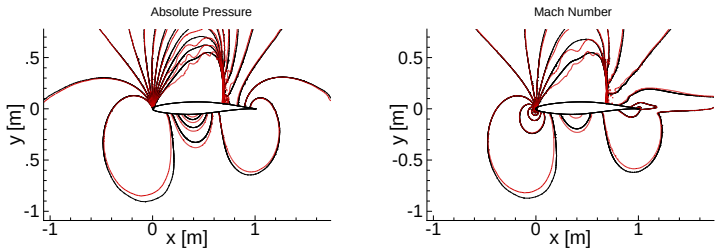
(f) Val. Case-6



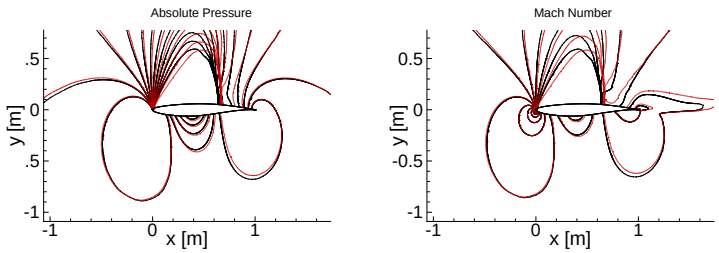
(g) Val. Case-7



(h) Val. Case-8

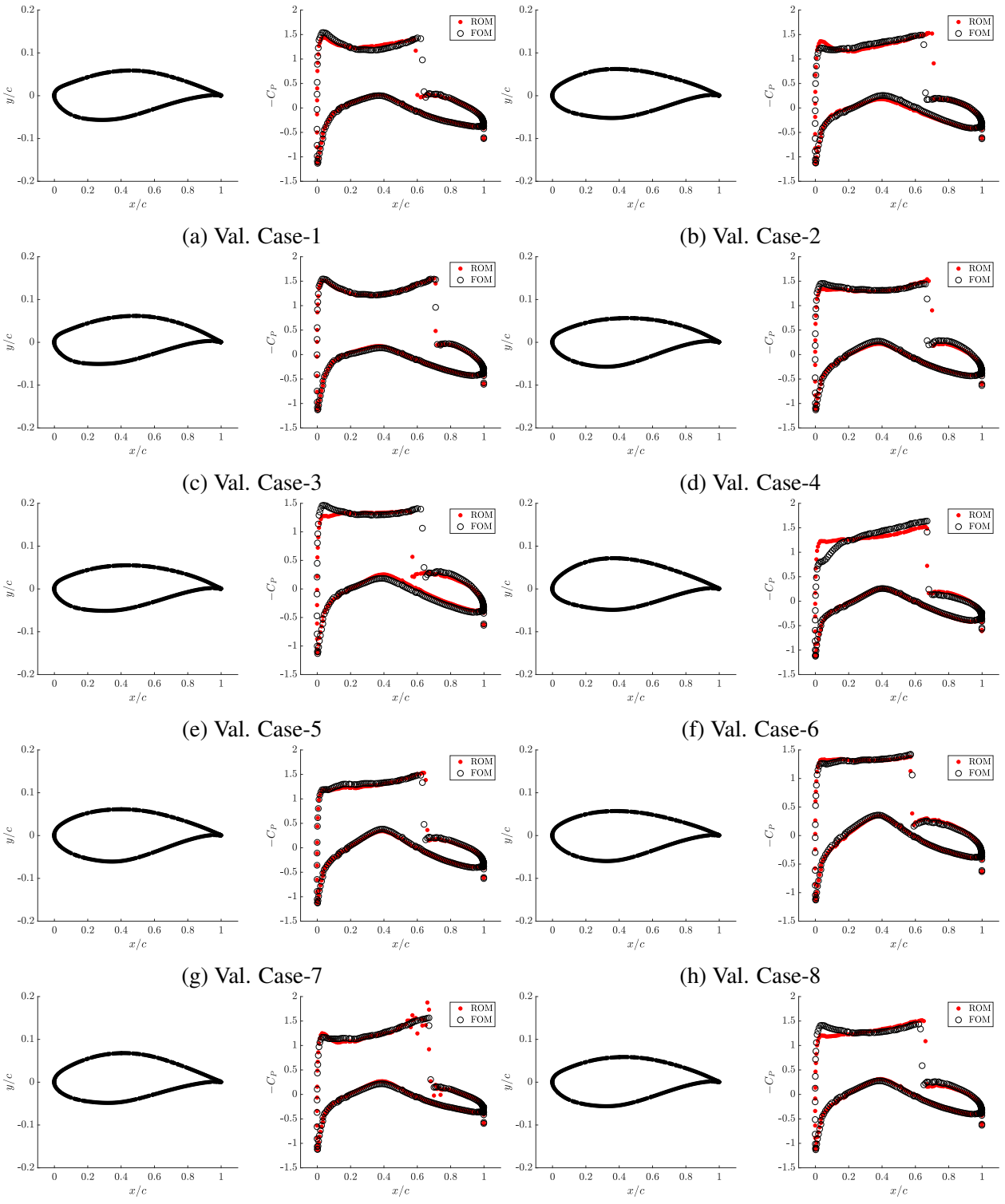


(i) Val. Case-9



(j) Val. Case-10

Figure 4.35: Comparison of Mach number and absolute pressure between ROM & FOM with respect to shape parameters for the NACA0012 test case



(a) Val. Case-1

(b) Val. Case-2

(c) Val. Case-3

(d) Val. Case-4

(e) Val. Case-5

(f) Val. Case-6

(g) Val. Case-7

(h) Val. Case-8

(i) Val. Case-9

(j) Val. Case-10

Figure 4.36: Comparison of pressure coefficient C_P predicted by the ROM with the true solution due to the FOM for various airfoil shapes that represent the validation cases

4.2.4 Discussion

Validation of the ROM methodology with shape parameters was performed. Under subsonic shock-free conditions for the NACA test case, the predictions were consistently under 5% leading to very high accuracy. However, under transonic conditions with moving shocks in the RAE2822 test case, the predictive capability drops leading to an average error of about 15%. Under discontinuities in the flow field such as moving shocks, a POD-based method such as the present method is unable to accurately predict shock location and strength. In this respect, the model performance with shape parameters are very similar to that with flow parameters. Additionally, more snapshots were required for both the NACA and RAE test cases simply because there are more parameters - 2 in the case of flow parameters vs 6-8 in the case of shape parameters. Given that the model is ultimately going to be applied to situations where there could be 1000s of function evaluations, such a budget of high-fidelity snapshots (160 in this case), is well justified.

Despite the limitations in the transonic regime, it has been observed that the ROM still predicts the shock within 5% chord-length variability. Further, the C_P and C_l are still predicted with an average error of 14% and 4% respectively. While the drag coefficient is predicted with an average error of 16%, due to its high sensitivity to discrepancy in pressure distributions, it incurs very large error in certain cases upto, 50%.Overall, the main observation is that the pressure distributions and lift coefficient are predicted with much better accuracy than drag coefficient, similar to the previous section.Therefore, the suitability of the method for design optimization in the transonic regime has additional challenges that need to be addressed before they can be applied in such contexts.

As observed in the previous section, the type of interpolation plays a very important role with shape parameters also. For the NACA test case, the $2nd$ order interpolation gave best results, while for the RAE test case the $1st$ order interpolation suited better; corroborating the previous remark that the best interpolation method for a given problem is highly problem dependent.

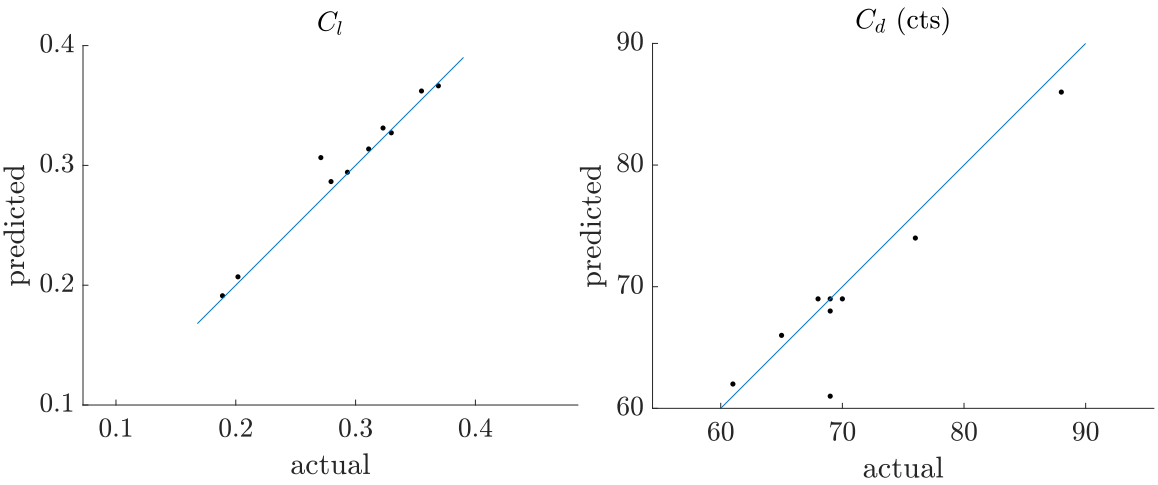
The overall prediction errors of the model are driven by the following main factors (i) POD basis truncation (ii) ROM interpolation errors (iii) DEIM interpolation errors and (iv) finite-precision round-off errors. Among these 4 components, (ii) is expected to dominate since the actual parametric dependence of the linear and non-linear terms are unknown and is also highly problem-dependent. Because of that *apriori* error bounds for the ROM error is a challenge and is not attempted in this thesis. However, based on the extensive validation process carried out in this work, the 3 outputs ($C_d,\ C_l,\ C_P$) are predicted within $5\ \%$ given adequate snapshots.

4.2.5 ROM Vs. Data-fits

It has been thus far demonstrated that the proposed methodology is able to capture the non-linear flow field by solving an approximate and reduced form of the governing equations. It should be noted that when only the (scalar-valued) system outputs are of interest, a typical off-the-shelf data-fit surrogate model could have the capability to approximate them as good as (if not better than) the ROM. This is demonstrated in Figure 4.37, where the data-fit surrogate is a $2nd$ order response surface of the form (see [4])

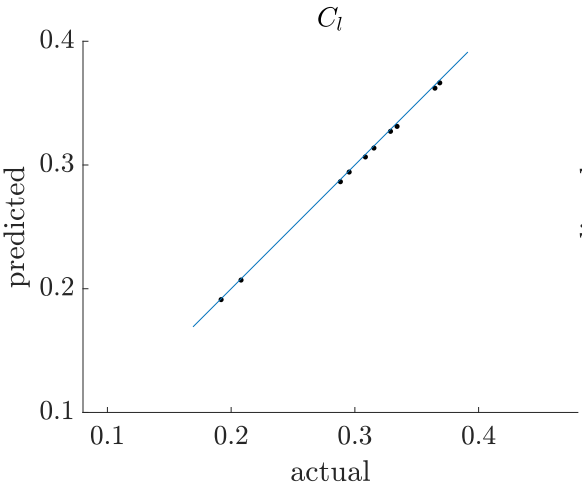
$$y = \beta_0 + \sum_{i=1}^{p} \sum_{j=1, j\neq i}^{p} \beta_{ij} x_i x_j$$

where the x_i, x_j represent the CST coefficients, β_0 and β_{ij} are the regression coefficients and y is the scalar response which is either C_d or C_l. Overall, from the data and the plots in Figure 4.37, it can be seen that the performance of both surrogates are similar, although the ROM predicts the drag coefficient slightly better than the response-surface.

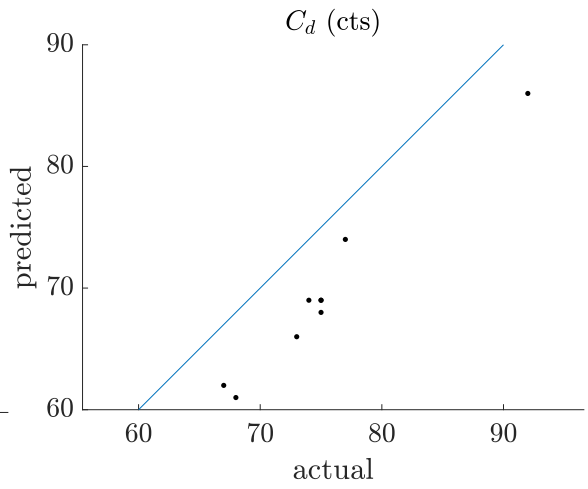


(a) ROM

(b) ROM



(c) Response Surface



(d) Response Surface

Figure 4.37: Comparison of scalar response prediction via ROM (a. and b.) and response surfaces (c. and d.). Notice that the prediction capabilities are quite similar between the two surrogate modeling approaches. However, data-fit surrogates (such as response surfaces) lack the ability to approximate field variables similar to the ROM.

In low-dimensional design spaces, where only the surrogate model of a scalar response is of interest, a data-fit surrogate model such as **response surfaces** performs as good as the ROM. However when **field variables** are of interest, a physics based approach such as the ROM is necessary.

4.3 Concluding Remarks

In this chapter, the ROM methodology is demonstrated on the compressible Euler equations which forms a coupled set of 4 non-linear parametric PDEs to capture inviscid compressible flow. Model validation is performed at different flow regimes (subsonic and transonic) and with both flow parameters and shape parameters. While the evaluation of the model so far was primarily based on accuracy, here 2 additional utilities of the method are emphasized.

4.3.1 Utility in computing flow Adjoints

The adjoint method offers an efficient approach to computing the sensitivities of an objective function with respect to parameters and states. Let there be an objective function $g(\mathbf{u}, \boldsymbol{\theta})$ and recall that $\mathbf{u}$ is the state variable and $\boldsymbol{\theta}$ are a set of parameters. Then adjoint method computes the total derivative $\frac{dg}{d\boldsymbol{\theta}} = \frac{\partial g}{\partial \boldsymbol{\theta}} + \frac{\partial g}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial \boldsymbol{\theta}}$ where the cost of computing the total derivative is independent of the dimensionality of $\boldsymbol{\theta}$ and hence the efficiency. However, computing $\frac{\partial \mathbf{u}}{\partial \boldsymbol{\theta}}$ is expensive since it requires computing the state variable via costly simulations.

To see this, let us assume for the sake of simplicity that $\mathbf{u}$ is the solution of a linear system, $\mathbf{A}\mathbf{u} = \mathbf{b}$. Then the derivative with respect to one component of $\boldsymbol{\theta}$, θ_i is $\mathbf{A}\mathbf{u}_{\theta_i} + \mathbf{A}_{\theta_i}\mathbf{u} = \mathbf{b}_{\theta_i}$, where the subscript refers to the variable with respect to which the derivative is calculated. So $\mathbf{u}_{\theta_i} = \mathbf{A}^{-1}\left(\mathbf{b}_{\theta_i} - \mathbf{A}_{\theta_i}\mathbf{u}\right)$ and overall $\mathbf{u}_{\boldsymbol{\theta}} = \mathbf{A}^{-1}\left(\mathbf{b}_{\boldsymbol{\theta}} - \mathbf{A}_{\boldsymbol{\theta}}\mathbf{u}\right)$. Now going back to the total differential $\frac{dg}{d\boldsymbol{\theta}} = g_{\boldsymbol{\theta}} + g_{\mathbf{u}}\mathbf{A}^{-1}\left(\mathbf{b}_{\boldsymbol{\theta}} - \mathbf{A}_{\boldsymbol{\theta}}\mathbf{u}\right)$, where the terms $\mathbf{b}_{\boldsymbol{\theta}}$ and $\mathbf{A}_{\boldsymbol{\theta}}$ can be computed via methods such as automatic differentiation, which are expensive.

However, the ROM offers utility here since the linear operator $\mathbf{A}$ is computed as part of the methodology. Secondly, the adjoints may be computed at the ROM level, which are a smaller system and hence more efficient. Therefore, with the proposed approach, in addition to predicting the field variables and outputs of non-linear flow-fields accurately, the adjoints of objective functions can also be cheaply calculated which finds application

in problems such as sensitivity analysis and optimization.

4.3.2 Utility in multi-fidelity approaches

It was earlier emphasized that the proposed approach being POD based, is limited in its ability to accurately predict flows with discontinuities. Further, any surrogate model is dependent on the high-fidelity model for its construction and typically assumes that the high-fidelity model is the truth. In other words, despite being a *physics-based* approach, the proposed approach depends on the high-fidelity model in the sense that the final reduced order model can only be as accurate as the high-fidelity model and can not extrapolate outside training data to predict new physics.

Therefore, the reduced order models, despite their capabilities, may not entirely be able to replace the high-fidelity models. However, they can be used in conjunction with the high-fidelity models itself such that the overall budget of high-fidelity simulations for a given application can be significantly reduced. Such an approach is called the *multi-fidelity* approach (see Chapter 1) where by applying domain knowledge, a certain *trust region* can be constructed where the ROM is known to make accurate predictions. This way, the expensive high-fidelity models are used only when necessary whereas the ROM is not depended upon to make accurate predictions everywhere in the design space.

CHAPTER 5

APPLICATION: DESIGN OPTIMIZATION & PROBABILISTIC ANALYSIS

We finally demonstrate the present methodology on the main motivation of this thesis presented in Chapter 1 - applications in many-query context which include design optimization and uncertainty quantification. Specifically for design optimization, we demonstrate the method on inverse design, shape optimization problems. Following that the method is demonstrated on the probabilistic analysis problem where we quantify the uncertainty in the shape parameters on the airfoil lift and drag force coefficients.

5.1 Inverse Design

Research Question 7. *How does the ROM perform on the inverse design problem?*

Under certain circumstances, a specific aerodynamic load distribution about an aerodynamic object is of interest. For instance, under incompressible flow assmptions for a finite wing, an elliptic lift distribution along the wing is known to produce the least induced drag [144]. Similarly, in the preliminary design of propellers [145] and turbines[146], a certain lift distribution along the blade is an input to the design process. In such cases the actual design (shape) parameters that produce such a load distribution is of interest. We call such a process the *inverse* design. Here we fix a certain pressure coefficient distribution as our *target* and search the design space for the shape parameters that would best approximate

Table 5.1: Free-stream conditions for inverse design

P_∞	101325 Pa
T_∞	288 K
ρ_∞	1.225 kg/m^3
a_∞	340.296 m/s
M_∞	0.6
α	2 $^\circ$

the target. Therefore we are interested in solving the following optimization problem

$$\begin{aligned} \underset{\theta}{\text{argmin}} \quad & \frac{1}{2}\|C_P(\mathbf{u},\boldsymbol{\theta}) - C_P^*(\mathbf{u},\boldsymbol{\theta})\|_2^2 \\ \text{subject to:} \quad & \\ & \mathbf{R}(\mathbf{u},\boldsymbol{\theta}) = 0 \\ & \boldsymbol{\theta}_l \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}_u \end{aligned} \tag{5.1}$$

Where C_P^* is the target pressure distribution. Naturally we want to replace the full-order governing equations with the ROM and hence we solve the modified problem

$$\begin{aligned} \underset{\theta}{\text{argmin}} \quad & \frac{1}{2}\|C_P(\mathbf{u},\boldsymbol{\theta}) - C_P^*(\mathbf{u},\boldsymbol{\theta})\|_2^2 \\ \text{subject to:} \quad & \\ & \Psi^T\mathbf{R}(\Phi\tilde{\mathbf{u}},\boldsymbol{\theta}) = 0 \\ & \boldsymbol{\theta}_l \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}_u \end{aligned} \tag{5.2}$$

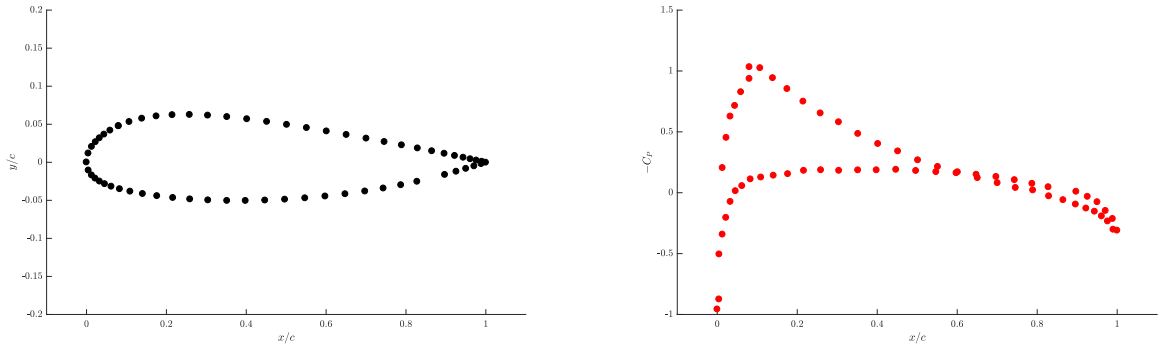


Figure 5.1: Target airfoil shape and pressure distribution

The free-stream conditions used for this test case are summarized in Table 5.1. The target C_P^* and the corresponding airfoil shape is shown in Figure 5.1. The optimum shape was searched using a Genetic Algorithm based optimizer with a population size of 30 per generation and a total of 60 generations. The constraint and function convergence tolerance were set to $1E-5$ and $1E-3$ respectively and the optimization required a total of 1830 function evaluations of the ROM , to determine the final design. Overall, the optimization required approximately 3.7 hrs of wall-clock time running in serial mode.

The convergence history of GA optimizer is shown in Figure 5.2; the fitness function ceased to improve significantly beyond 60 generations. There is a discrepancy (in terms of relative error) of about 4.4% between the target and predicted pressure distributions. Overall, the ROM is able to predict the airfoil shape within a small error margin as shown in Figure 5.3. Further, it does so within wall-clock time of approximately 3.7 hrs whereas the equivalent FOM wall-clock time for the same number of function evaluations is expected to take roughly 300 hrs. Therefore the computational efficiency achieved via the ROM outweighs the rather small ($< 5\%$) error in its prediction. Having said that, certain amount of error in the prediction of the ROM is always to be expected - this is due to several factors including (i) POD basis trunction (ii) a finite snapshot size (iii) ROM interpolation error and (iv) the lack of explicit treatment of boundary conditions. Overall, this test case demonstrates the usefulness of the ROM for a design optimization setting where it enables fast decision making.

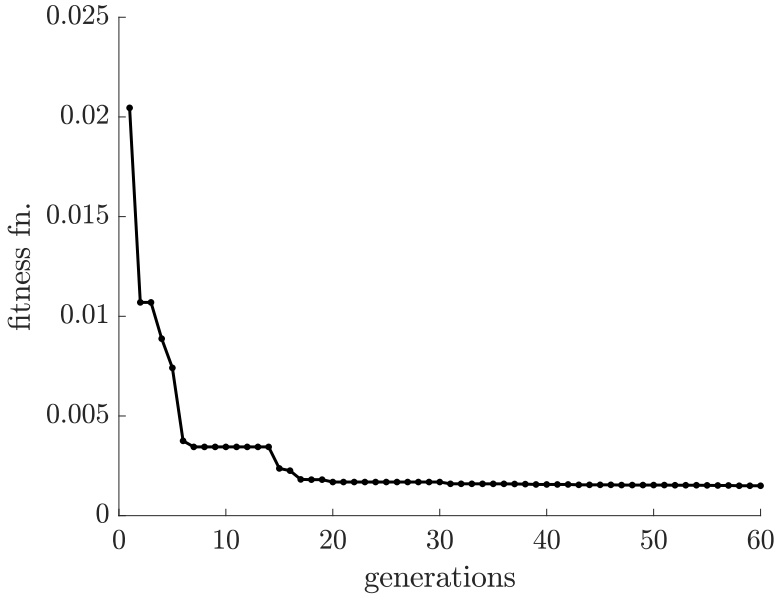
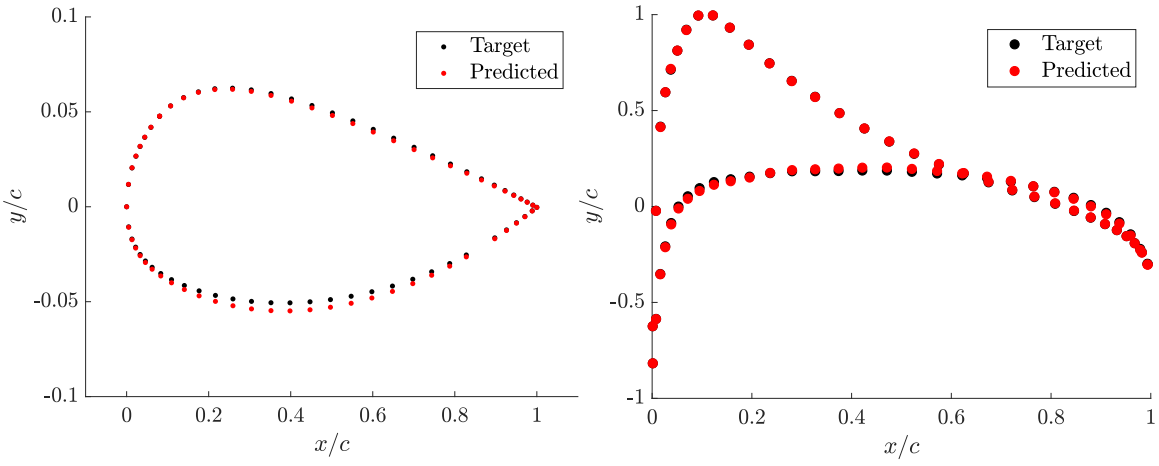


Figure 5.2: Optimizer convergence



(a) Airfoil shape comparison

(b) C_P distribution comparison

Figure 5.3: Comparison of predicted-target design (via ROM) with the actual target

5.2 Aerodynamic Shape Optimization

Research Question 8. *How does the ROM perform on the Aerodynamic Shape Optimization (ASO) problem?*

Aerodynamic Shape Optimization (ASO) enables the exploration of novel design alternatives for wing shapes from an aerodynamic efficiency point of view. However, an effective application of ASO requires the appropriate parametrization, variable bounds and constraints for a given problem. Additionally, ASO is a high-dimensional [138], multi-modal optimization problem where the multi-modality is mainly driven by the need for using as many design degrees of freedom as possible [139]. The multi-modality aspect is very well addressed by gradient-free, global optimization methods, such as the Genetic Algorithm (GA) [140], Particle Swarm Optimization (PSO) [141] or Simulated Annealing (SA) [142] just to name a few. Such methods typically do not depend on local gradients but on tune-able parameters that control their exploration and exploitation properties which aid in the determination of the global optimum. However, under a high-dimensional design space such an approach might be infeasible when the cost of the objective function (and constraint) evaluations are prohibitively expensive. In such situations, an accurate ROM fills the gap and aids in global optimization of expensive functions. While gradient-based methods such as the *adjoint-based* [76] methods are very effective in high-dimensional problems [138, 79, 78], the known issue of multi-modality in ASO [143], could lead to the optimizer getting stuck at a local optima. Indeed, there have been reports suggesting that unless a gradient-free optimization is not possible, a hybrid of both methods (gradient & gradient-free) methods is necessary to ensure the globally optimum design is found [143].

The ROMs previously constructed with shape parameters in Chapter 4 are now used in an optimization setting. The goal here is to search for the globally minimum drag configuration within the design space. The flow parameters ($\mathbb{M}, \alpha$) are held fixed, while the CST coefficients are treated as the design variables. ROMs are then used in the objective

function and constraint evaluations as described below. The capability of the methodology to be applied to global optimization with constraints is demonstrated with the NACA0012 test case introduced in Chapter 4. For comparison, the true optimum is determined based on running the FOM at a dense grid of the parameters with 1000 samples. To accomplish this specific task, computing resource on a cluster with parallel processing and enhanced processing power (relative to the workstation on which the ROM is evaluated) was used which required a total of 10 hrs of wall-clock time. Although such a large budget of FOM evaluations are not in general feasible for practical problems, given the small size of the test problems used in this work, this was deemed acceptable. Therefore they serve as the validation for the optimum predicted by the ROM.

5.2.1 Problem Statement

We solve the following PDE-constrained optimization problem

$$\begin{aligned} \underset{\theta}{\text{argmin}} \quad & \delta\, C_d(\mathbf{u}, \boldsymbol{\theta})^2 \\ \text{subject to:} & \\ & \mathbf{R}(\mathbf{u}, \boldsymbol{\theta}) = 0 \\ & \mathbf{h}(\mathbf{u}, \boldsymbol{\theta}) = 0 \\ & \mathbf{g}(\mathbf{u}, \boldsymbol{\theta}) \leq 0 \\ & \boldsymbol{\theta}_l \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}_u \end{aligned} \tag{5.3}$$

where $\delta \in \mathbb{R}$ is an arbitrary value used to condition the problem[1], $\boldsymbol{\theta}$ represents the CST coefficients that are the design variables of the problem with upper bound $\boldsymbol{\theta}_u$ and lower bound $\boldsymbol{\theta}_l$. $\mathbf{h}$ and $\mathbf{g}$ represent the equality and inequality constraints respectively. The $\mathbf{R}(\mathbf{u}, \boldsymbol{\theta}) = 0$ is the FOM (state-space model) that needs to be solved to compute the

[1]For instance, if it is known that $C_d(\boldsymbol{\theta}^*)$ is in $\mathcal{O}(10^{-3})$, then δ may be chosen as 10^6 such that the objective function is in $\mathcal{O}(1)$. This way, other parameters such as the convergence tolerance can be specified relatively in an appropriate manner.

objective function at every iteration of the optimization.

The above problem is replaced with the ROM as follows

$$\begin{aligned} \underset{\boldsymbol{\theta}}{\text{argmin}} \quad & \delta\, C_d(\Phi\tilde{\mathbf{u}}, \boldsymbol{\theta})^2 \\ \text{subject to:} & \\ & \Psi^T \mathbf{R}(\Phi\tilde{\mathbf{u}}, \boldsymbol{\theta}) = 0 \\ & \mathbf{h}(\Phi\tilde{\mathbf{u}}, \boldsymbol{\theta}) = 0 \\ & \mathbf{g}(\Phi\tilde{\mathbf{u}}, \boldsymbol{\theta}) < 0 \\ & \boldsymbol{\theta}_l \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}_u \end{aligned} \tag{5.4}$$

where the term $\Phi^T \mathbf{R}(\Phi\tilde{\mathbf{u}}, \boldsymbol{\theta}) = 0$ represents the ROM, Φ and Ψ are the trial and test basis set respectively (see Chapter 1). Note that the objective function and constraints now depend only on the reduced state vector ($\tilde{\mathbf{u}}$) and hence the dimensionality of the problem has been reduced. Further the evaluation of the outputs (the objective function and constraints in this case) at every optimizer iteration depends only on a small subset of points that fall on the surface of the geometry; see Appendix A for further details. The optimization is tested for 2 types of problems that are generalized by Equation 5.4 namely (i) with only bound constraints - which are inequality constraints on the shape parameters and, (ii) with C_l constraint - which is a non-linear equality constraint.

5.2.2 Optimization with bound constraints

The optimization problem is re-stated below as follows

$$\begin{aligned} \underset{\boldsymbol{\theta}}{\text{argmin}} \quad & \delta\, C_d(\Phi\tilde{\mathbf{u}}, \boldsymbol{\theta})^2 \\ \text{subject to:} & \\ & \Psi^T \mathbf{R}(\Phi\tilde{\mathbf{u}}, \boldsymbol{\theta}) = 0 \\ & \boldsymbol{\theta}_l \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}_u \end{aligned} \tag{5.5}$$

Table 5.2: Freestream Conditions and design variable bounds for the NACA0012 test case

		$\boldsymbol{\theta}_l$	$\boldsymbol{\theta}_u$
P_∞	101325 Pa	0.1182	0.2196
T_∞	288 K	0.1889	0.3509
ρ_∞	1.225 kg/m^3	0.0971	0.1803
a_∞	340.296 m/s	-0.2196	-0.1182
$\mathbb{M}$	0.6	-0.3509	-0.1889
α	2 $^\circ$	-0.1803	-0.0971

Table 5.3: Genetic Algorithm Tuning parameters

Number of generations	N_{gen}	30 & 60
Population Size	P_{gen}	30
Mutation Probability	μ	0.8
Cross-over type	–	Scattered
Cross-over fraction	–	0.8
Mutation type	–	Uniform

where, θ_l and θ_u represent the $\pm 30\%$ bounds of the baseline CST coefficients. The freestream conditions and the design variable bounds are listed in Table 5.2. The scalar δ in Equation 5.5 is set to be $1E + 6$ in order to keep the objective function roughly in $\mathcal{O}(1)$ as it approaches the optimum.

A global optimization is carried out using Genetic Algorithm with the settings listed in Table 5.3 and the optimization history is shown in Figure 5.5. A population size of 30 samples per generation was used and the convergence tolerance was set to $1e - 3$ for the objective function and $1e - 5$ for the constraints. The optimization converged after 30 generations leading to a total of 930 function calls (the extra 30 function calls is to generate the initial population).

The optimized shape and C_P distributions are shown in Figure 5.6. Since the flow is at an angle of attack, the optimizer tends to flatten the lower surface of the airfoil to dampen the gradients near the leading edge that mainly contributes to drag. From the experience of the author, the drag coefficient is very sensitive to changes in pressure at the leading edge for subsonic cases, and therefore the optimizer tending to cause most change around this region of the airfoil (and in the bottom surface since the flow is arriving at an incidence

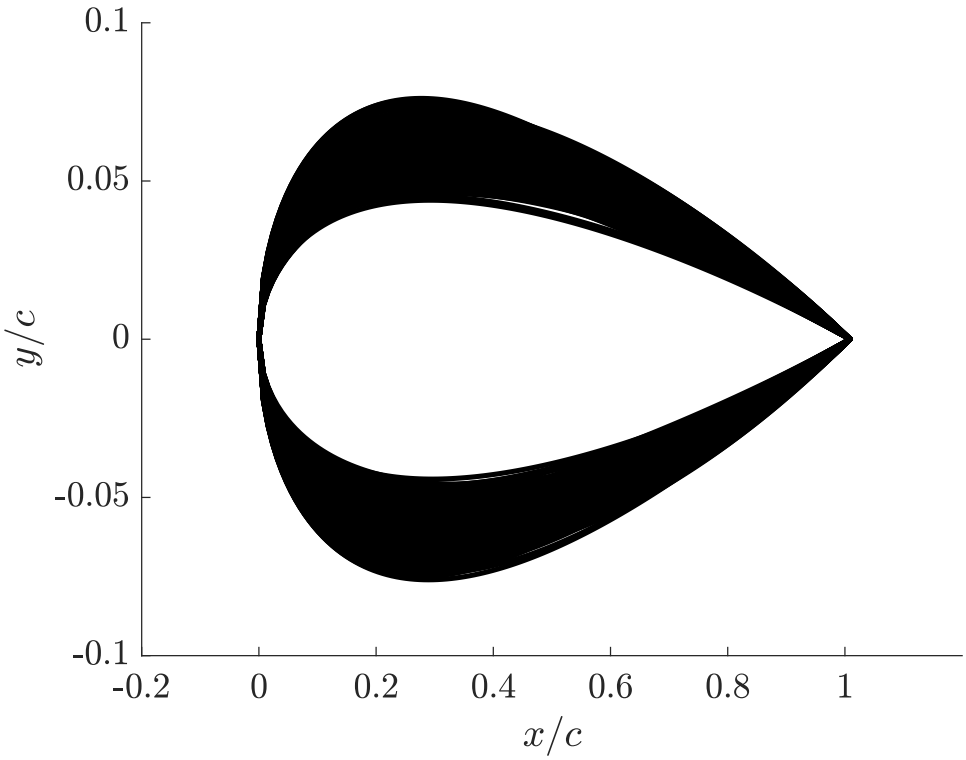


Figure 5.4: Airfoil design space generated by perturbing the NACA0012 baseline by $\pm 30\%$.

angle) is physically justified. Further, in Figure 5.6b it can be seen that the suction peak is lower than the baseline value leading to the decrease in drag.

We further compare the ROM results against the FOM solution obtained by running the optimum shape (as predicted by the ROM) in the high-fidelity model. The pressure coefficient is shown in Figure 5.8, where the agreement of the ROM results with the FOM is evident. A similar comparison is also made in Figure 5.7 in terms of the absolute pressure and mach number; a testimony of the capability of the ROM methodology to accurately capture the physics of the underlying system at a fraction of the computational cost.

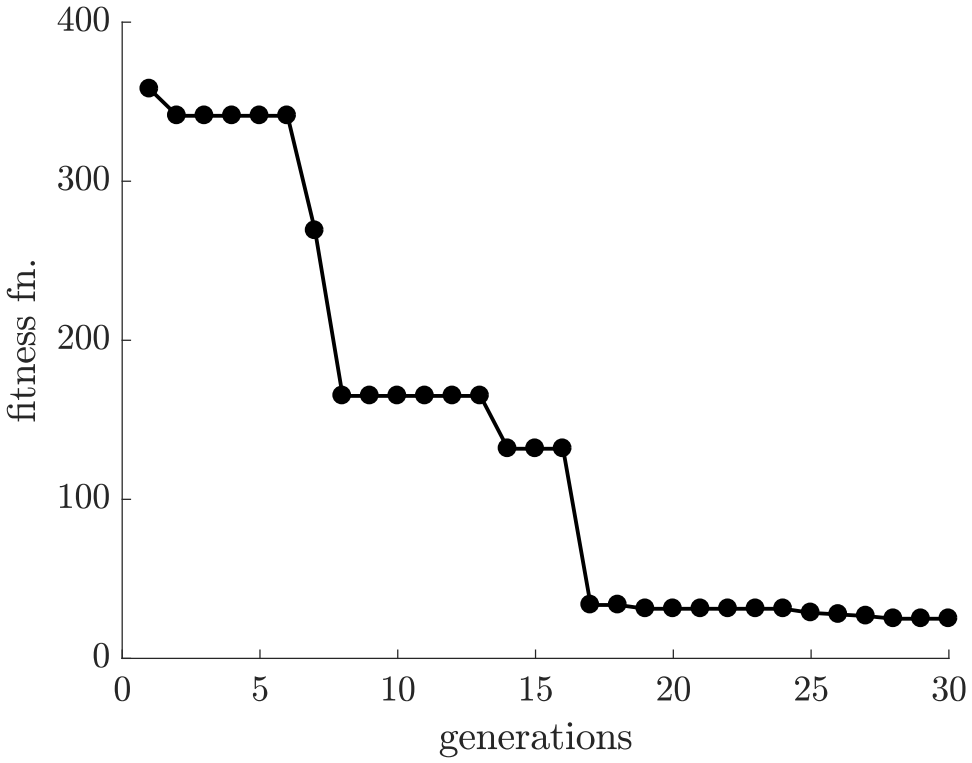
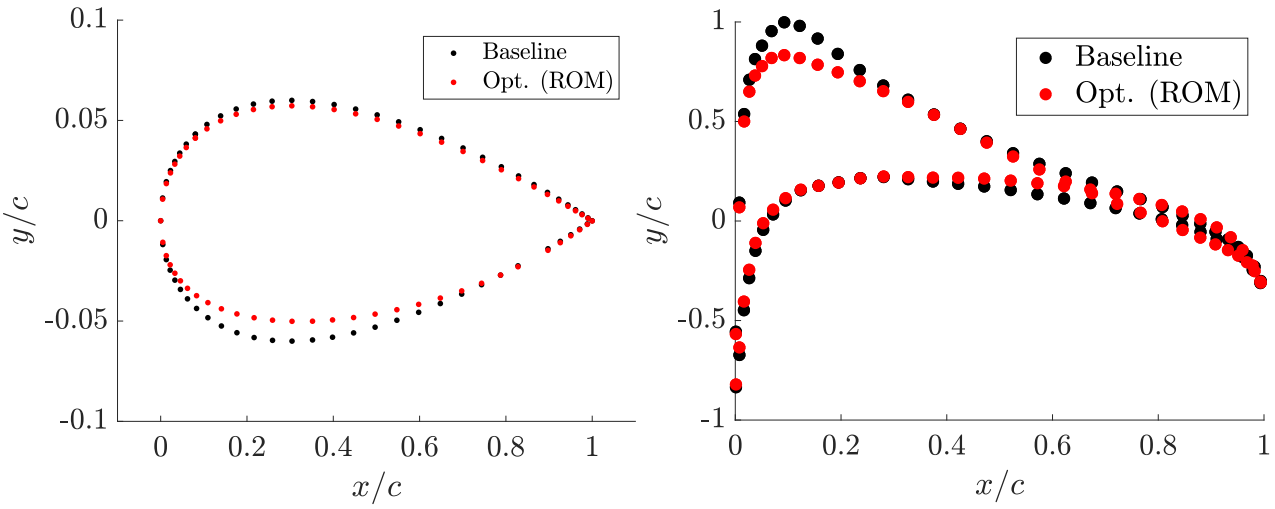


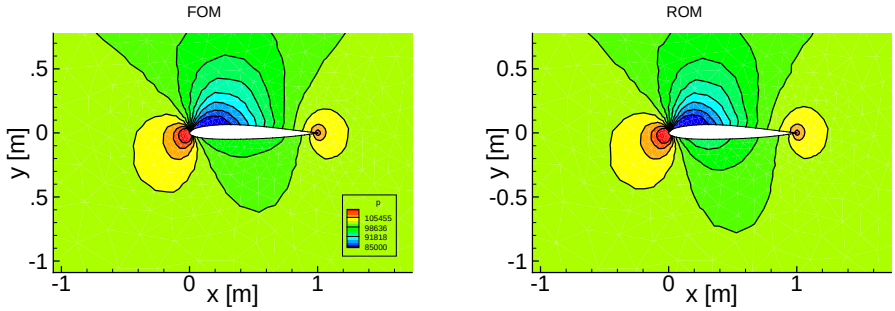
Figure 5.5: GA optimization history for the NACA test case



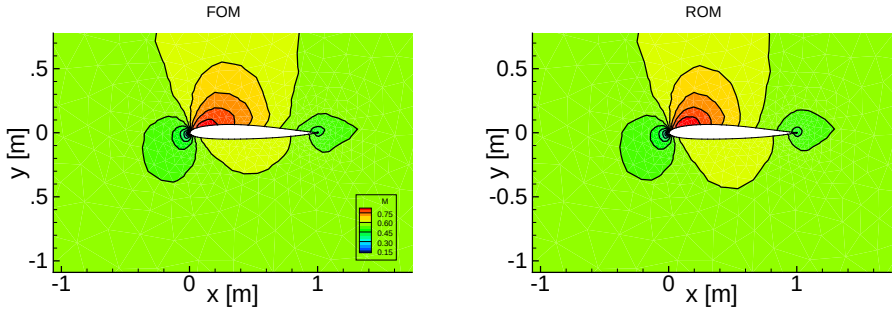
(a) Optimized airfoil shape Vs baseline

(b) Optimized airfoil C_P Vs baseline

Figure 5.6: Comparison of the optimum against the baseline



(a) Absolute Pressure



(b) Mach Number

Figure 5.7: Comparison of the global optimum predicted by the ROM (right) against the high-fidelity solution (ROM)

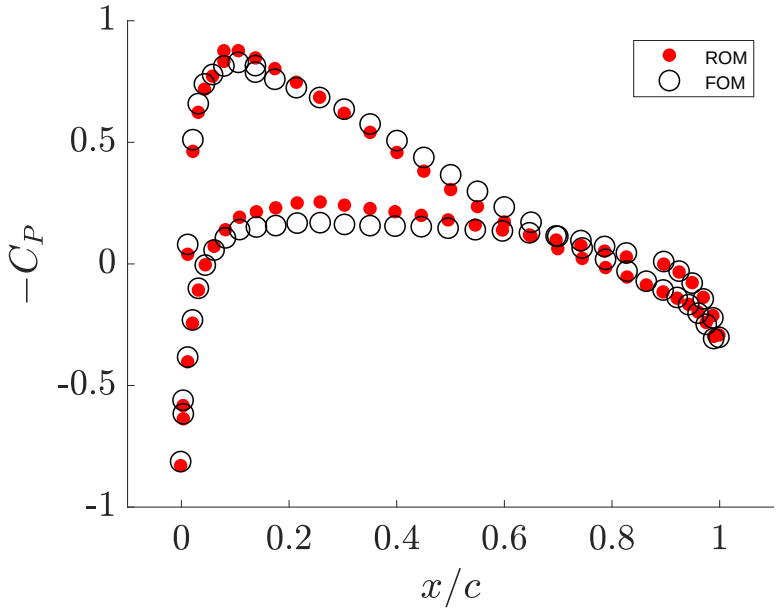


Figure 5.8: Optimized airfoil and C_P distributions. ROM Vs FOM

In order to ensure that the predicted optimum is truly the global optimum within the

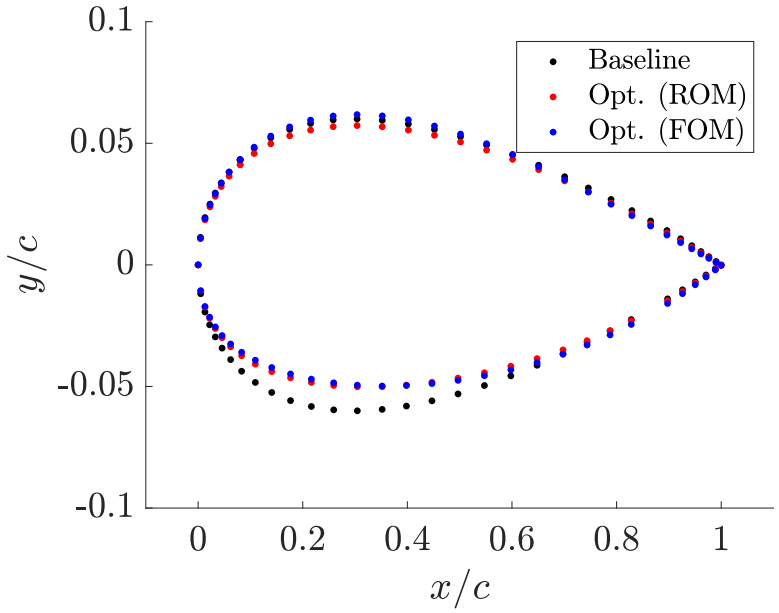


Figure 5.9: Optimized airfoil shapes

chosen design space, a well defined method of verification process was necessary. In this regard, a dense DOE consisting of 1000 points in the design space was generated and the FOM was evaluated at each point. Note that the truly global optimum is a result of a well-balanced and thorough exploration and exploitation of the design space. However by densely sampling the design space, we expect that approximate location of the global optimum can be estimated. Additionally, this provides a visualization of the design space that could offer insight into the influence of each design variable.

Firstly, the true global optimum (based on the 1000 sample dense DOE) is compared against the ROM prediction in Figure 5.9. The ROM and FOM optimum agree very well on the bottom surface of the airfoil, which is where most of the shape deformation has occurred for this test case. On the top surface however, there is discrepancy and this is explained using visualizations of the full design space in the following passages.

The Figure 5.10 shows a scatter plot of the DOE used to run the high-fidelity simulations. Each grey circle represents one high-fidelity sample. The larger black circle represents the global optimum based on the dense FOM samples and the large green circle represents the global optimum as predicted by the ROM via GA optimization. Notice

that the ROM results are in the neighborhood of the FOM results. In terms of the CST2 coefficient, the ROM prediction has the highest discrepancy with that of the FOM. Due to the fidelity difference between the ROM and the FOM, one would not expect the ROM to exactly predict the global optimum consistent with the FOM. However, the reason for the discrepancy was further investigated.

As mentioned earlier, the drag computation is known to be extremely sensitive to the pressure distribution. The solution predicted by the ROM is known to incur round-off errors in the projection step due to the large-scale matrix multiplication in finite precision. Therefore, it should be expected that the prediction of the ROM (in terms of drag count) has atleast a few drag counts of uncertainty.

While there is no estimate of this uncertainty, the designs that were within 3 drag counts of the global (FOM-predicted) optimum were identified. This is shown in Figure 5.11, where these points are identified as the black circles (smaller in size than the optimum identified in the previous figure). Large spread in the design variable for a relatively small 3 drag counts is quite evident. In this plot, it can be seen that the ROM prediction (green circle) is within the spread of designs that are within 3 drag counts of the global optimum. While this does not necessarily prove that the design space of this optimization problem is multi-modal, it does suggest that the global optimum is not necessarily significantly better than several other local optima. Given such an uncertainty about the model, the ROM-based approach leads to an optimum that is within that range. Additionally, in the figure, it can be seen that the CST4 is the only design variable that has the most impact on the design variable, since all the points are clustered towards the constraint boundary. Further, the ROM prediction also falls in the same localized neighborhood - the constraint boundary. Overall, it can be said that the ROM-based optimization gives a *useful* design which is in the neighborhood of the global optimum.

Finally, the ROM prediction comes after 930 function evaluation which required approximately 2 hours of wall-clock time. The same budget of function calls in the FOM

(with identical convergence criteria and serial computing) would require roughly 155 hours of wall-clock time. Therefore, the computational gains from the ROM is compelling, while it does trade a small amount of accuracy for a relatively large gain in efficiency.

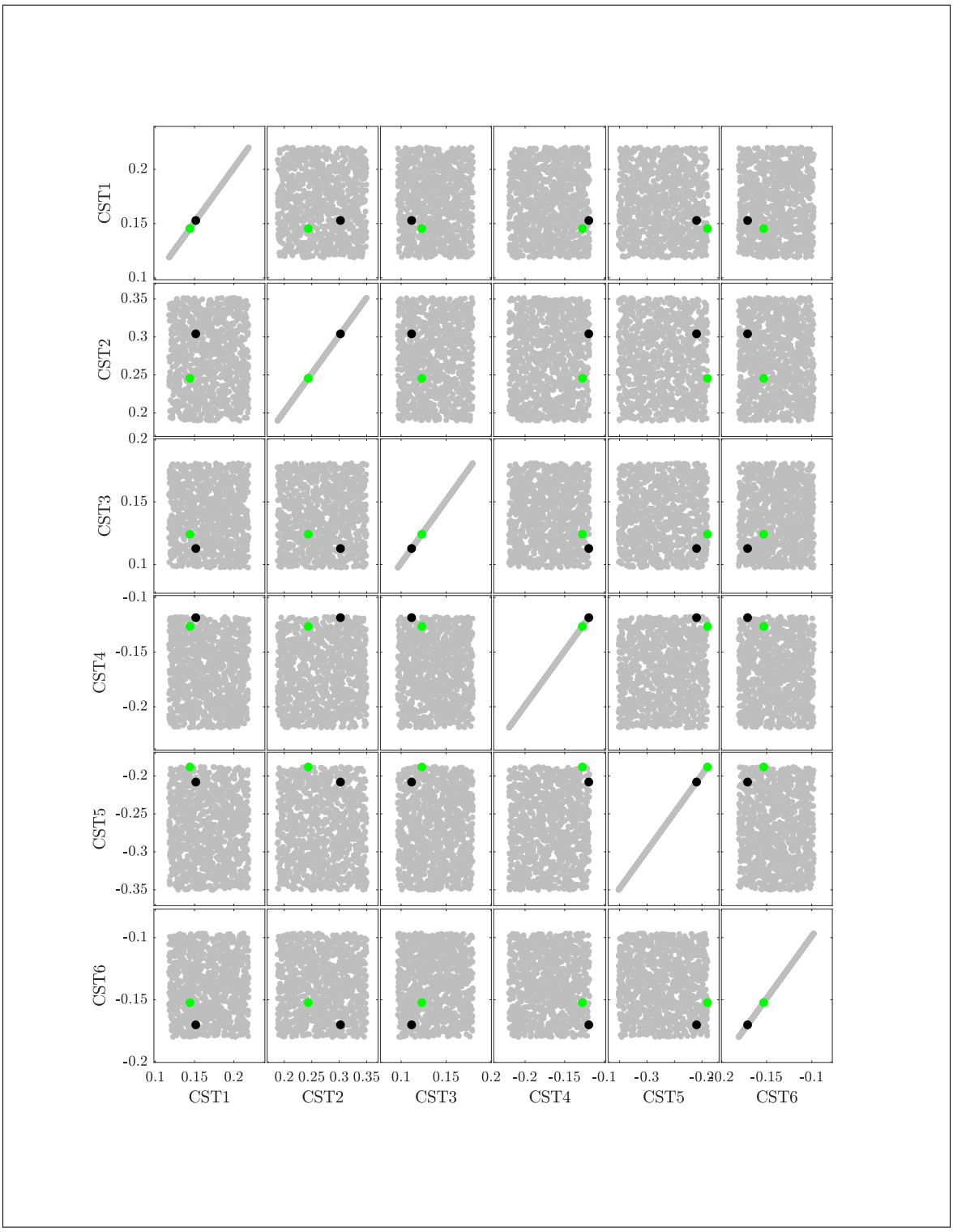


Figure 5.10: Green - global optimum based on 1000 high-fidelity simulations. Black - global optimum determined using the ROM. Grey - 1000 high-fidelity simulation samples.

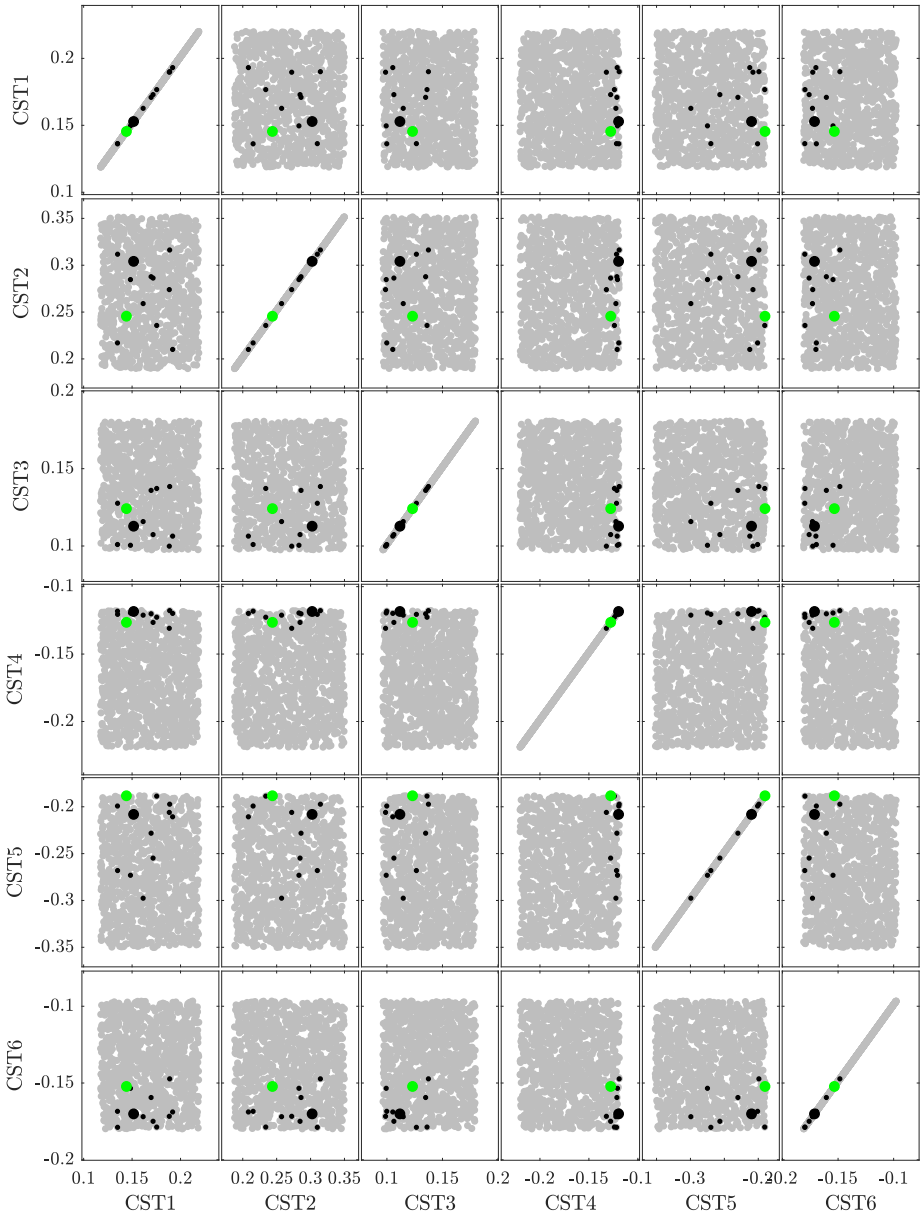


Figure 5.11: Green - global optimum based on 1000 high-fidelity simulations. Black - global optimum determined using the ROM. Grey - 1000 high-fidelity simulation samples.The smaller black circles denote points within 3 drag counts of the minimum C_d among the high-fidelity samples.

5.2.3 Optimization with bound & C_l constraint

We now repeat the optimization problem with an equality constraint in terms of the lift coefficient. We test the capability to the ROM to find the global optimum while being constrained to a certain target lift coefficient, C_l^{target}. The constraint is added as a penalty term and the optimization problem is stated as follows

$$\begin{aligned} \underset{\boldsymbol{\theta}}{\text{argmin}} \quad & \delta_1 \, C_d(\Phi\tilde{\mathbf{u}}, \boldsymbol{\theta})^2 + \delta_2 \, \left(C_l(\Phi\tilde{\mathbf{u}}, \boldsymbol{\theta}) - C_l^{target}\right)^2 \\ & \text{subject to:} \\ & \Psi^T \mathbf{R}(\Phi\tilde{\mathbf{u}}, \boldsymbol{\theta}) = 0 \\ & \boldsymbol{\theta}_l \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}_u \end{aligned} \tag{5.6}$$

The $C_l^{target} = 0.43$ and the scalars δ_1 and δ_2 were set to $1e + 6$ and the GA was run for 60 generations with a population size of 30 samples per generation. The optimizer history is shown in Figure 5.12 and stopped showing any significant improvement in the fitness function value after converging on C_l^{target} within 3 decimal places. The optimizer airfoil shape is shown in Figure 5.13; notice the bottom surface of the airfoil showing similar trends as the previous case (without the C_l constraint) due to the angle of attack in the flow. Further, camber is added to the airfoil shape in order to meet the lift constraint.

A comparison with the true global optimum (from 1000 high-fidelity samples) is shown in Figure 5.14. The deformation in the bottom surface of the airfoil predicted by the ROM almost exactly matches with the true optimum. Further, the upper surface deformation as predicted by the ROM shows a similar trend as the true optimum, although there is discrepancy. The scatter plot shown in Figure 5.14 compares the ROM-predicted optimum with the true optimum - notice that they lie in close proximity to each other.

To investigate further, we check for variability in the C_l values of the high-fidelity data within $\pm 1\%$ of the C_l^{target} and C_d values within 3 drag counts of the true global optimum.

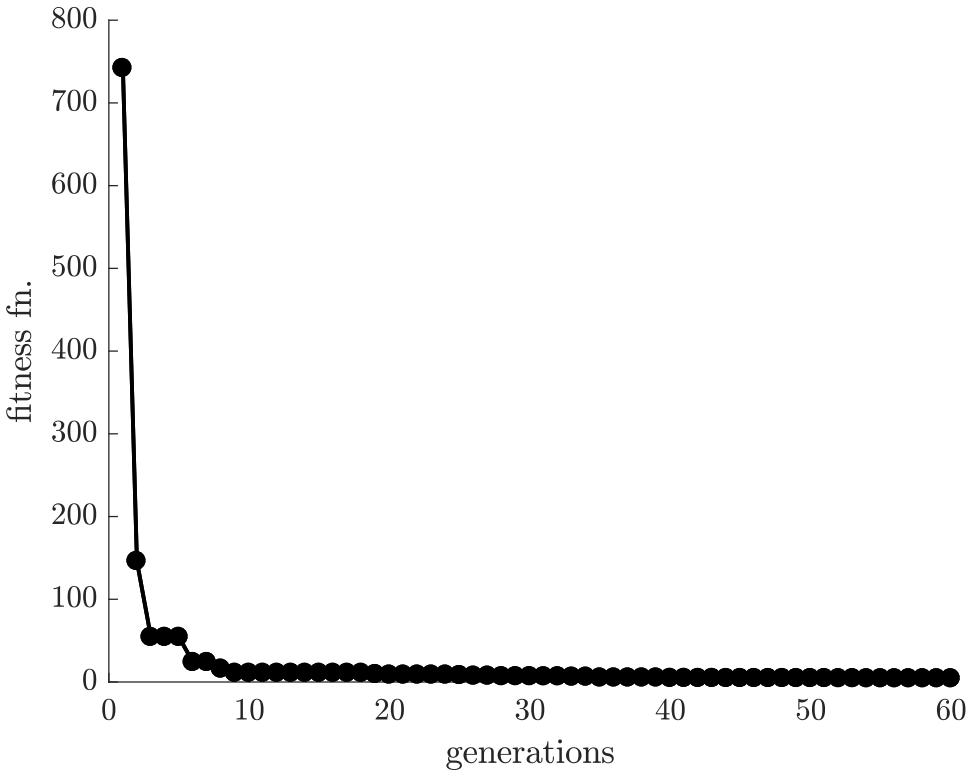


Figure 5.12: GA optimization history for the NACA-2 test case

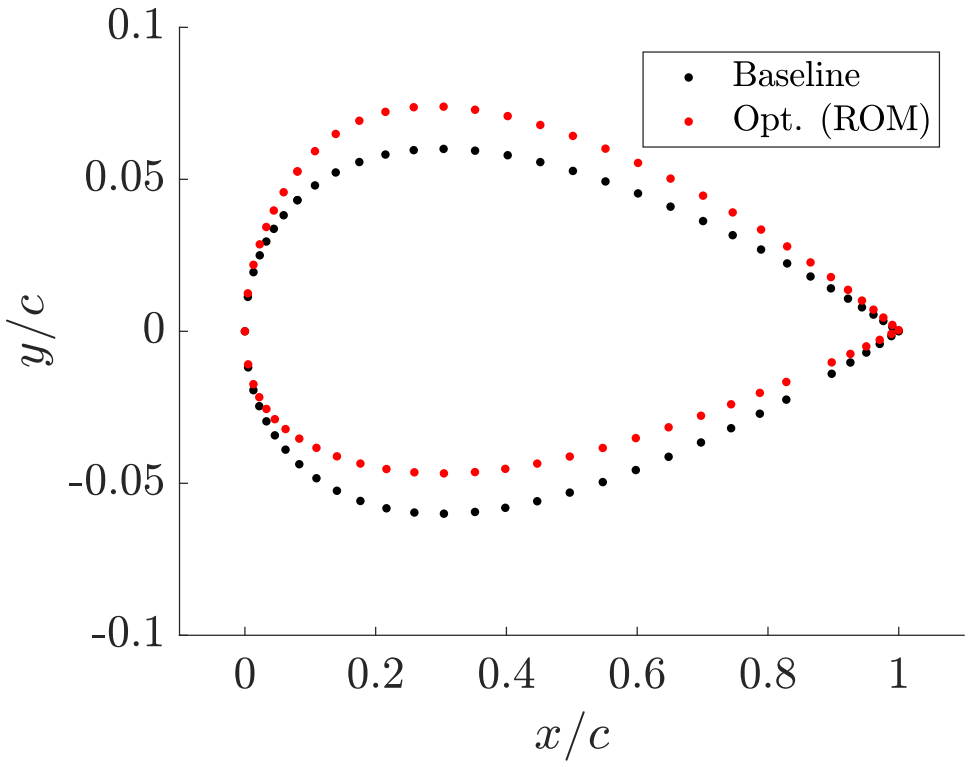


Figure 5.13: Optimized airfoil shapes

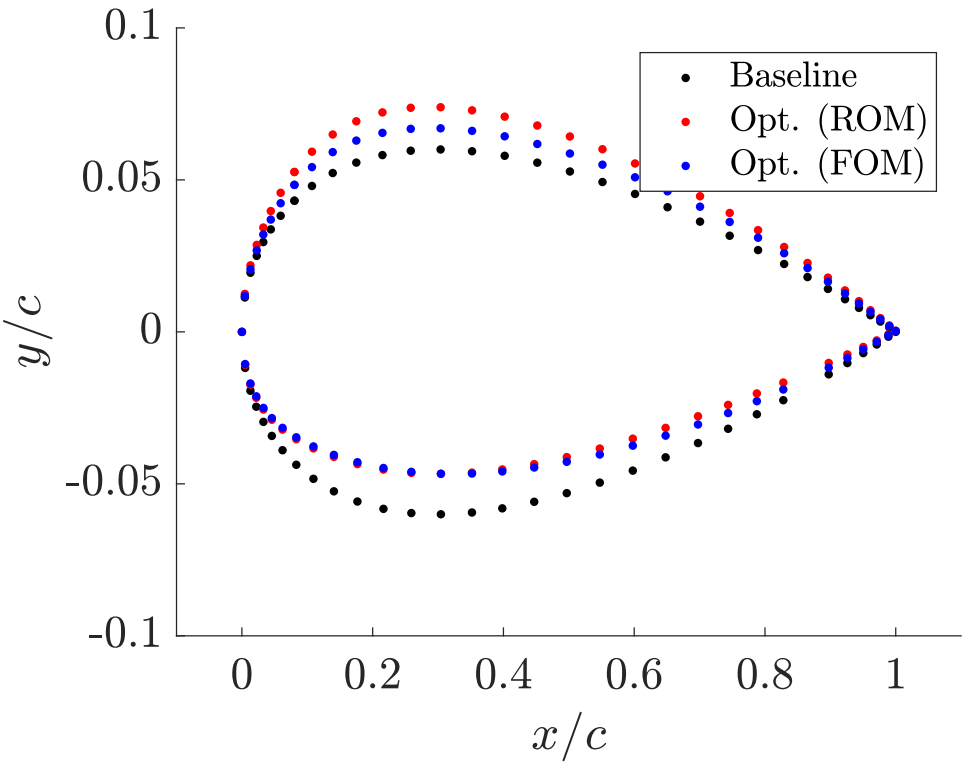


Figure 5.14: Optimized airfoil shapes

This is shown in Figure 5.16. There are fewer points within this variability range mainly because the variability in C_l is much less compared to that of C_d, as explained earlier. This plot shows that the ROM-predicted optimum is well within this variability range for all the CST coefficients but $CST6$. This discrepancy is the main contributor for the shape difference pointed out in Figure 5.14. Note that, there is inherent error in the prediction of the ROM as demonstrated through the validation efforts in Chapter 4; therefore one cannot expect that the ROM is able to arrive at exactly the same optimum as the high-fidelity model. However, the test cases so far show that the ROM produces a *useful* result that finds an optimum that is within a small localized neighborhood of the true optimum.

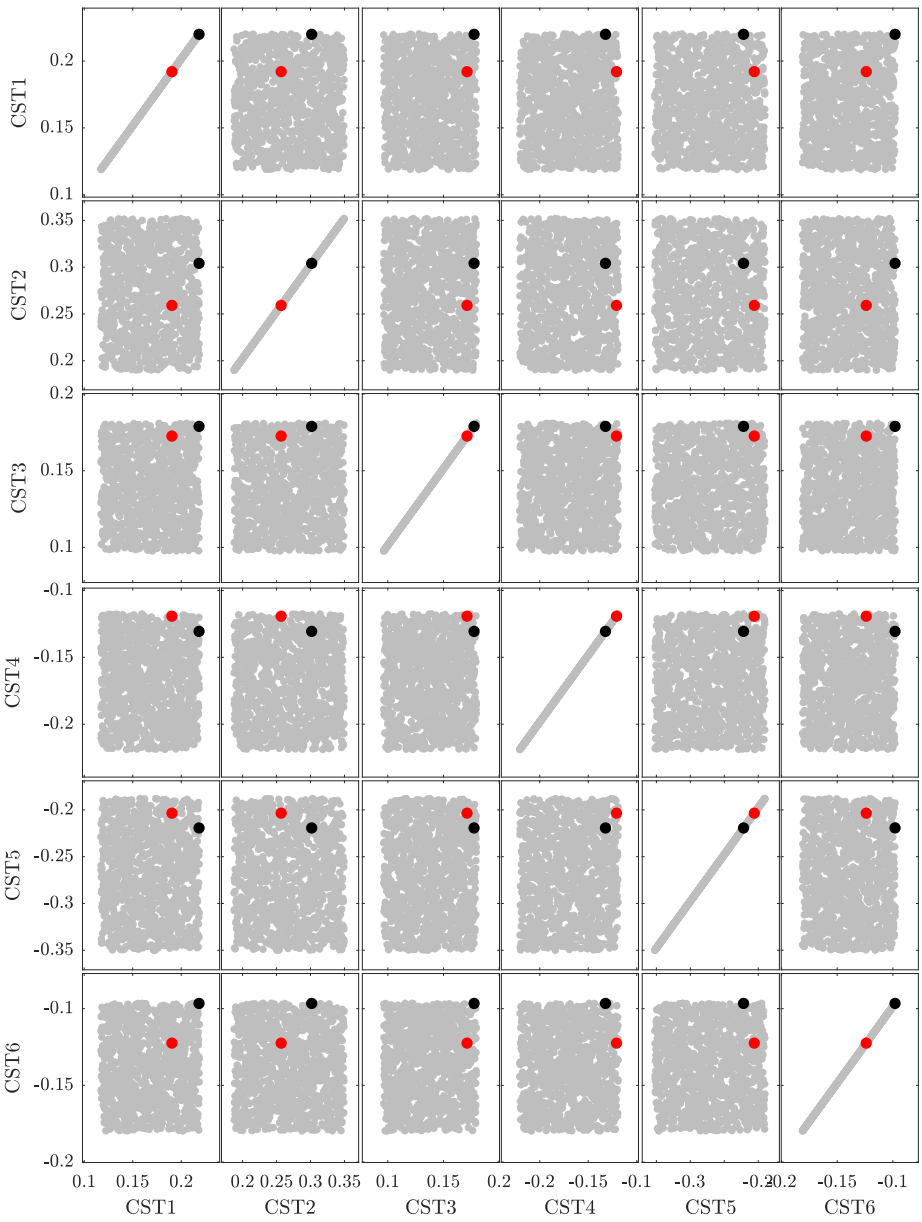


Figure 5.15: Black - global optimum based on 1000 high-fidelity simulations. Red - global optimum determined using the ROM. Grey - 1000 high-fidelity simulation samples.

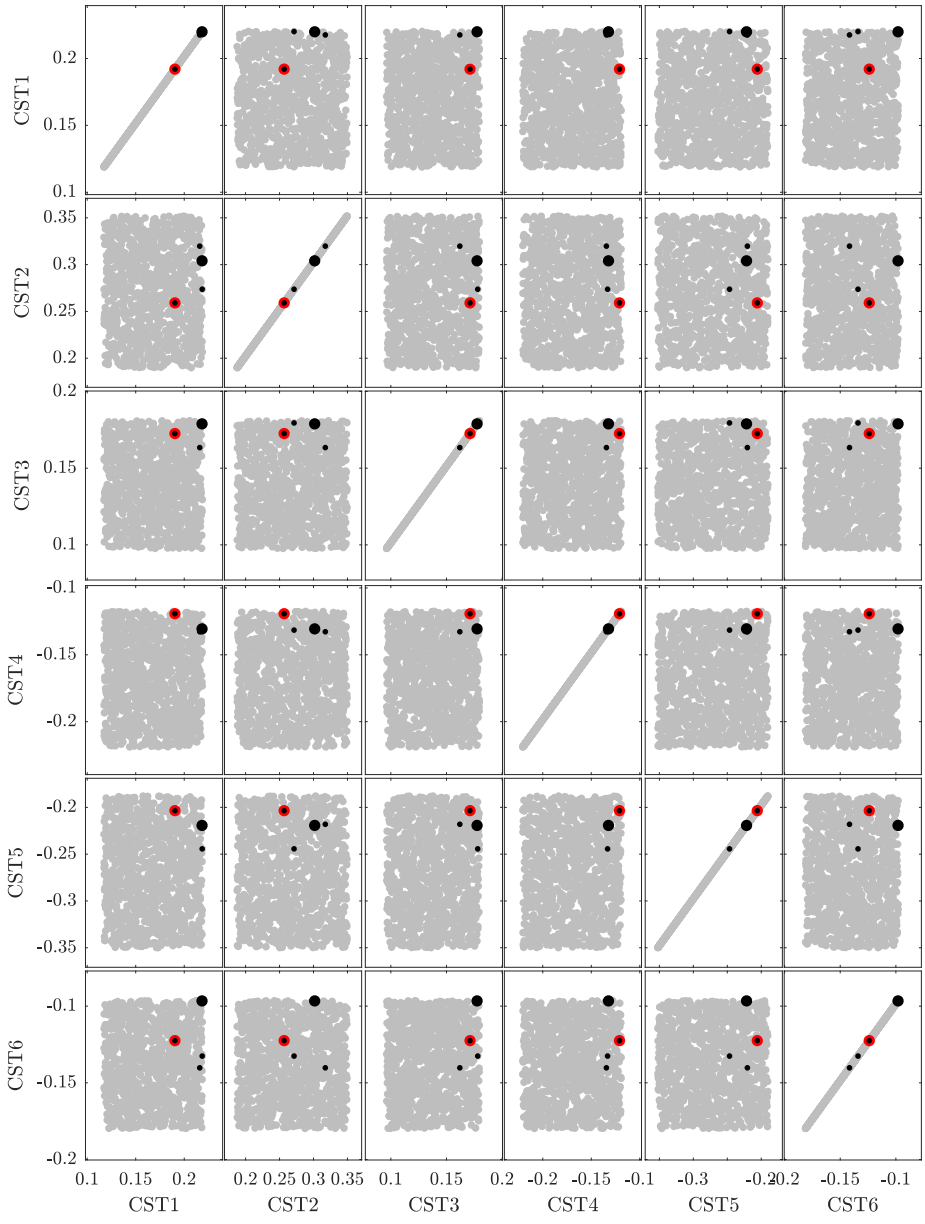


Figure 5.16: Black - global optimum based on 1000 high-fidelity simulations. Red - global optimum determined using the ROM. Grey - 1000 high-fidelity simulation samples. The smaller black circles denote points within $\pm 1\%$ of C_l^{target} and 3 drag counts of the minimum C_d among the high-fidelity samples.

5.3 Probabilistic Analysis

Research Question 9. *How does the ROM perform on the probabilistic analysis problem?*

Next, we demonstrate the application of the present methodology towards a probabilistic analysis. Uncertainty in design is inevitable and the decision making process should account for this uncertainty in order to make more reliable decisions early on in the design phase. Specifically, we are interested in quantifying the uncertainty in the airfoil lift and drag coefficients due to the manufacturing process induced variations in the airfoil shape. We use the same $\pm\ 30\%$ variation in the airfoil CST coefficients and uniformly sample this design space; see Figure 5.17. A Monte Carlo [147] simulation is carried out to propagate the input uncertainty into the ROM in order to quantify the uncertainties in the system outputs - namely, the lift and drag coefficients.

The histograms of the responses are shown in Figures 5.18 and 5.19. The ROM histograms are approximated from 4000 samples run via the Monte Carlo simulations. These simulations were run in serial on a desktop computer which consumed approximately 7.2 hrs of wall-clock time. An equivalent budget of high-fidelity simulations run in serial for the same level of convergence as the ROM would have cost $\approx$ 667 hrs of wall-clock time; therefore the computational speed-up with the ROM is $\sim\ \ 100\times$. The FOM histograms were generated from 1000 latin hyper cube samples that were used in the previous section. While these are not-exact one:one comparisons and also represent approximations from a rather small sample size, they are used to make high-level observations about the similarities between the predictions of the ROM and FOM. Firstly, the histogram for C_l is strikingly similar between the two with a mean of $\approx\ \ 0.29$ in both cases. The histogram looks approximately symmetric with a normal-like distribution. In the case of C_d the ROM predicts the right-ward skew of the distribution accurately. The range predicted by the ROM is approximately $[0.005, 0.016]$ while in the case of the FOM it is a bit narrower,

$[0.006, 0.01]$. The relatively higher sensitivity in the C_d computations compared to C_l, as explained in the previous sections, is the main reason for this discrepancy. Further, the number of points with $C_d > 0.01$ fall in the tail end of the distribution and form small fraction of the total number of samples; which can be considered as outliers. Similarly a small fraction of the points fall below 0.006 (lower bound predicted by the FOM) for the same reason. Secondly, the FOM histograms are constructed from a rather coarse(1000) sample of points and hence an exact comparison would require at the very least an equal sample size for both the ROM and FOM. While this might be feasible for the present test case (given its relatively cheaper computational cost when executed with parallel computing) it is avoided because the idea is not scalable for pratical problems. We compare them with the only goal of making a high-level observation about the ROM's capability to capture the actual trends of the FOM. Overall, the ROM is able to capture the generalized trends in the distributions of the response with useful accuracy.

We go a step further and compare the statistics of the distributions predicted by the ROM with that of the FOM. These are summarized in Table 5.5. As for the C_d, the ROM predicts the mean and median very close to the FOM, while the remaining statistics show greater discrepancy. The main reason is attributed to the presence of outliers in the C_d data, most of which is likely to be due to the higher sensitivity of this output to variations in C_P - a common thread observed across all results in this thesis. However, in the case of C_l, the predictions are much closer between the ROM and FOM. Overall, the ROM is able to capture general trends such as the range of outputs and shapes of their distributions with *useful* accuracy. One needs to be careful in comparing the ROM results against the FOM in Table 5.5 because both results are an outcome of samples of different sizes while both being a relatively coarse dataset for accurate approximations of distributions.

Table 5.4: Comparison of output (C_d) statistics, ROM Vs FOM

	ROM	FOM
Mean	0.0076	0.0075
Median	0.0076	0.0074
Std. Dev	0.0018	0.0007
Skewness	0.2294	0.3887
Kurtosis	4.5908	2.5210

Table 5.5: Comparison of output (C_l) statistics, ROM Vs FOM

	ROM	FOM
Mean	0.2996	0.2980
Median	0.2989	0.2969
Std. Dev	0.0587	0.0598
Skewness	0.0743	0.0431
Kurtosis	2.4794	2.7697

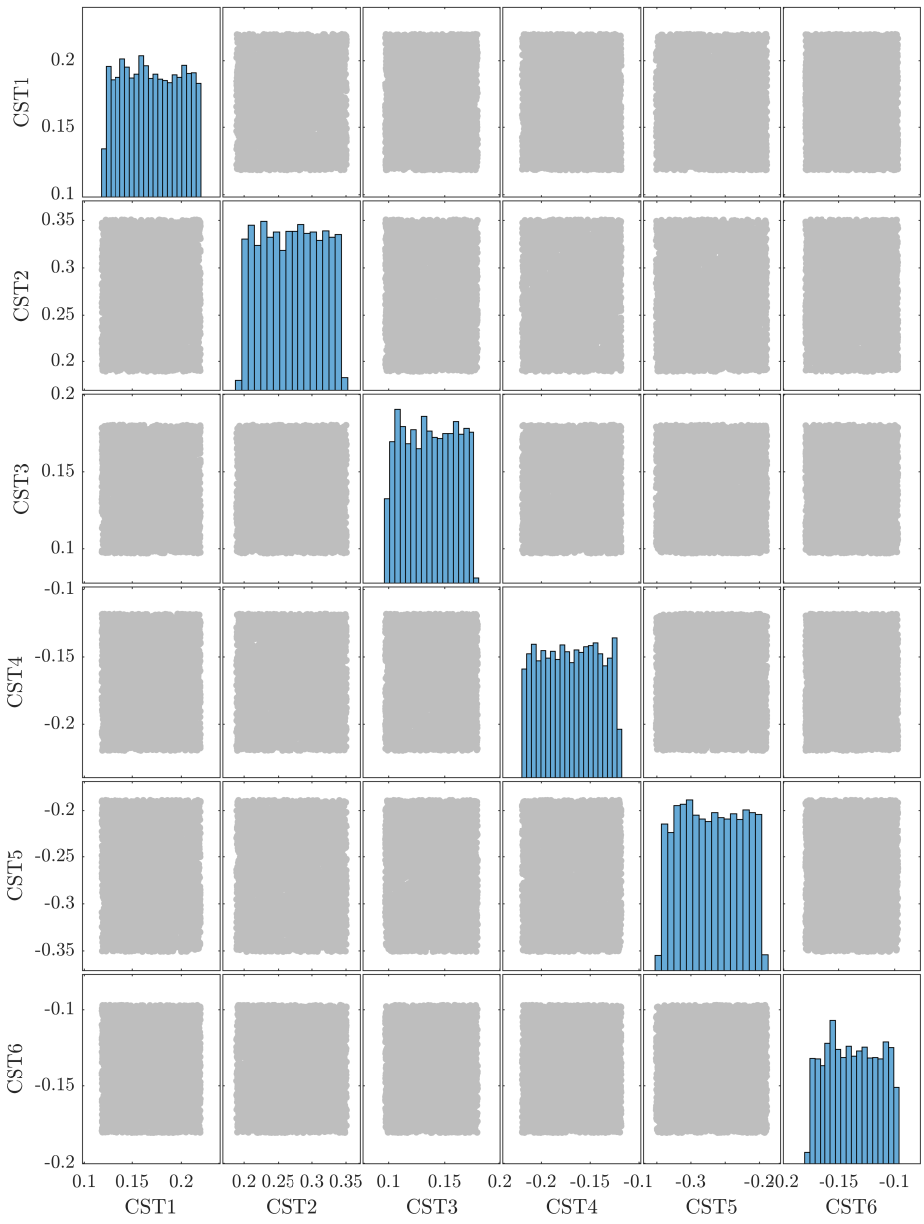
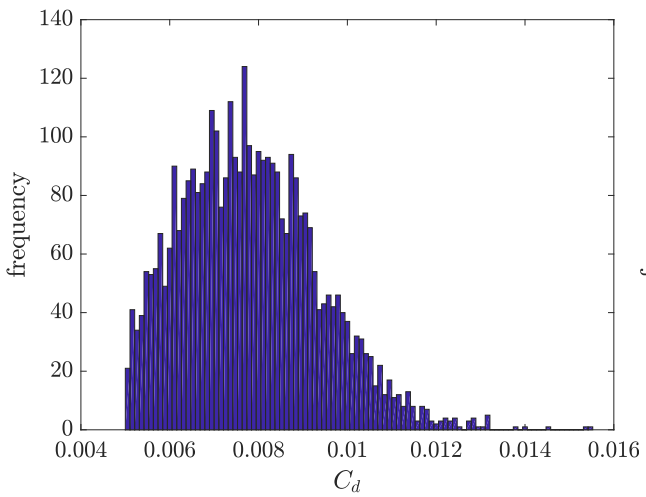
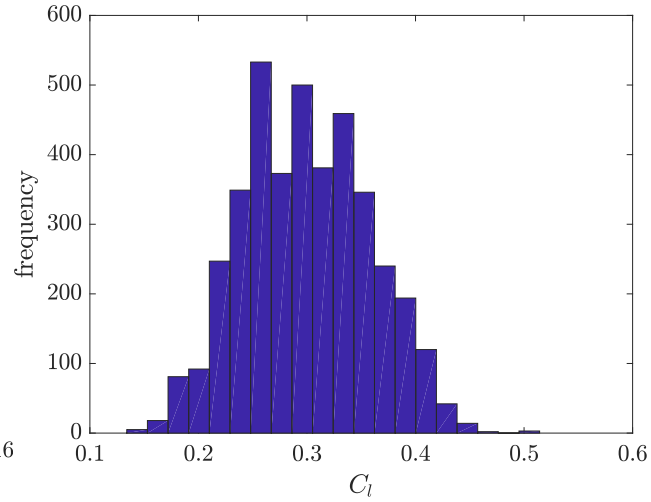


Figure 5.17: Uniformly distributed input CST coefficients for the probabilistic analysis

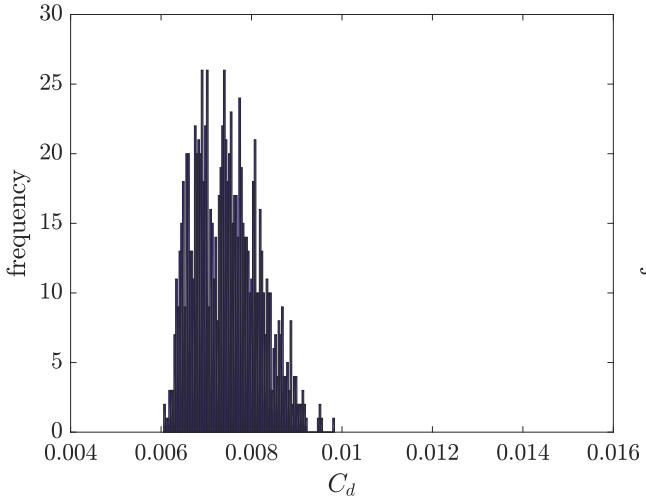


(a) Drag coefficient

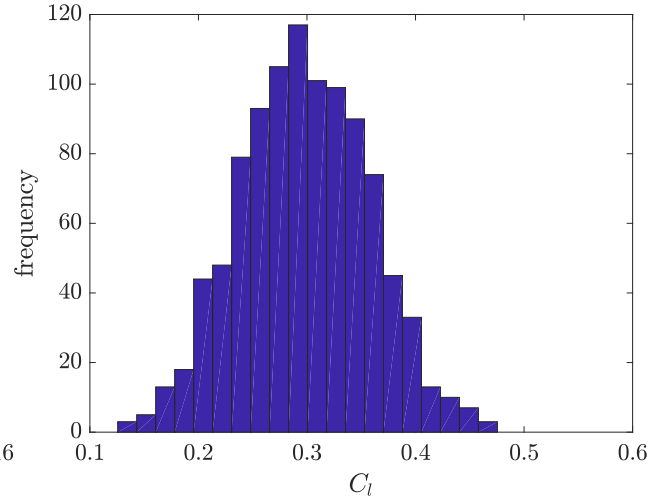


(b) Lift coefficient

Figure 5.18: Histograms via the ROM. 4000 monte-carlo samples.



(a) Drag coefficient



(b) Lift coefficient

Figure 5.19: Histograms via the FOM. 1000 LHC samples

5.4 Discussion

The method is applied to 3 specific applications in the many-query context: (i) aerodynamic shape optimization, (ii) inverse design and (iii) uncertainty quantification via Monte Carlo analysis. In all 3 situations, the ROM was executed $\mathcal{O}(1000)$ times at a wall-clock time of 2-8 hrs, while the equivalent budget of FOM would have consumed 200-800 hrs. Therefore

first and foremost, the results of this chapter demonstrate the utility of the methodology towards real-time decision making. Note that the solution method for the ROM currently does not take advantage of advanced computing resources such as parallel computing which are expected to significantly enhance the efficiency of the methodology even further.

In the aerodynamic shape optimization test case, the method was tested with and without a C_l constraint (in addition to design variable bound constraints). In both cases, the ROM was able to arrive at the a small neighborhood within the true global optimum (as determined from a dense sample of high-fidelity simulations). Further, the discrepancy observed in the predictions were identified to be mainly due to some variability in the drag coefficient and the ROM predictions were observed to be always within this variability. Finally, the ROM prediction of the true optimum design was simulated in the high-fidelity model and the comparison showed very good match between the two results. Overall, the ROM was able to achieve such accuracy at a fraction of the cost of the FOM.

The inverse design test case was mainly the test of the ROM to satisfy the physics of the problem, even in its approximated form. Given, a certain pressure distribution as target, the ROM was used to search the design space to identify the corresponding airfoil shape. The predictions of the ROM was within 5% of the actual solution while again, achieving it at superior computational efficiency compared to the FOM.

Lastly, a Monte Carlo analysis with 4000 uniformly sampled points from the input space was used to approximated the probability distributions of the two main outputs considered in this thesis: the C_d and C_l. The predictions were compared against a relatively coarse, 1000 sample FOM-based approximation. Overall, the C_l showed better match with the FOM results, while the C_d predicted the statistics with relatively higher discrepancy. Regardless, the predictions of the ROM turn out to be useful to make reliable decisions at a fraction of the cost of the FOM, which is its main overall goal.

CHAPTER 6
CONCLUSIONS & FUTURE WORK

A methodology to develop projection-based reduced order models with black-box high-fidelity PDE-based models is developed in this thesis. Specifically, non-linear, static, parametric systems are the main focus that find wide spread applications in aerospace design. The methodology is adequately validated under different parameter settings, model hyper-parameters are identified and the overall strengths and weaknesses of the approach are demonstrated. Finally, the model is applied to several many-query context problems described in Chapter 1.

6.1 General feasibility

The feasibility of the method is tested via canonical PDE test problems of the linear and non-linear type, each with 1 and 2 parameters respectively. The Matlab PDE toolbox which uses finite element based solver that was used as the black-box model that provides the snapshots. Overall, the ROM method lead to a dimensionality reduction of ≈ 26 times leading to several orders of magnitude of computational speed-up. The model accuracy for the linear test case was close to machine precision, while for the non-linear case it was $\sim 2\%$ on average. While these test cases are trivial from a practical value point-of-view, they allowed a feasibility test of the method within a computational cheap framework.

6.2 Model development & validation

From canonical PDEs in Chapter 3, we move to the compressible Euler equations in Chapter 4. These equations find numerous applications in the design of aerospace vehicle and represent a simplified form of the Navier Stokes equations. This chapter is dedicated to-

wards evaluating the model under two types of parameters: (i) flow parameters and (ii) shape parameters. The NACA0012 and the RAE2822 airfoils were used as test cases under subsonic and transonic flow regimes respectively.

6.2.1 Flow parameters

The flow parameters considered are the mach number, $\mathbb{M}$ and angle of attack, α. The major conclusions are summarized as follows

- Under subsonic conditions, the method is able to predict the state variables in $\mathcal{O}(1)\%$ accuracy
- Under subsonic conditions, the method is able to predict C_P, C_l and C_d consistently within 5%.
- Under transonic conditions, the method is able to predict the state variables in $\mathcal{O}(5)\%$ accuracy
- Under transonic conditions, the C_P is predicted within 15% error, C_l within 10% while the C_d incurs greater error in $\sim 20\%$
- Under transonic conditions, the method predicts the shock location within 5% chord lengths, provided there are adequate snapshots
- Under transonic conditions, C_d prediction accuracies of $< 5\%$ is achievable with a dense distribution of snapshots. However, similar accuracy in C_P and C_l are achievable with fewer snapshots
- The main hyper-parameters of the model are identified to be (i) choice of initial guess for the ROM and (ii) order of interpolation for the ROM. Using a *standardized euclidean* distance based nearest neighbor as initial guess, gave best results

for all test cases. The choice of the right interpolation is problem-dependent and is not known apriori.

- SInce the approach assumes that the state always lies in a linear hyper-plane constructed out of a globally valid basis, it is not suitable to extrapolate outside of the training design space.

6.2.2 Shape parameters

Next, the model is tested with airfoil shape parameters. Airfoil shapes are parameterized using Class-Shape Transformation (CST) functions whose coefficients are treated as design variables. The NACA airfoil is parameterized using 6 design variables while the RAE airfoil using 8. In the presence of shape parameters, the ROM is interpolated on the tangent space to the manifold in which they are embedded. The main conclusions are summarized as follows

- With shape parameters, more snapshots are required mainly because there are more parameters
- Under subsonic conditions, the results were consistent with what was observed with flow parameters - $< 5\%$ error across all outputs
- Under transonic conditions, greater error was incurred, but relatively better than flow parameters. This is mainly because within the chosen parameter range, the flow features do not change drastically.
- Under transonic conditions, the error in C_d was relatively higher than that in C_P and C_l; an observation consistent with results of flow parameters.

6.3 Application to design optimization

In Chapter 5, the method was applied to 3 types of problems: (i) aerodynamic shape optimization, (ii) inverse aerodynamic design and (iii) uncertainty quantification via Monte Carlo analysis. For applications (i) and (ii) a gradient-free global optimization approach was used. In each of these cases the total budget of function calls was in $\mathcal{O}(1000)$. The main conclusions of this chapter are summarized as follows

- For the ASO problems, the ROM predicts the true global optimum within 3 drag counts. The discrepancy is mainly driven by the fact that the C_d calculation was highly sensitive to errors in C_P and that the design space contained several designs within a small margin of 3 drag counts.
- For the inverse design problem, the ROM predicts the correct airfoil shape for a given C_P distribution with an error margin of 4%.
- For both the ASO and inverse design problems, the optimum designs were run using the high-fidelity model to confirm that the results match
- For the probabilistic analysis test problem, the ROM predicts the distribution statistics for C_l with much greater accuracy than C_d; again a general trend observed throughout this thesis
- For all the test problems, the total budget of function calls to the ROM was achieved at a wall-clock time $100\times$ faster than the same budget for the high-fidelity model

6.4 Known limitations

The following are some known limitations of the present methodology

- The appropriate ROM interpolation order is not known apriori. An inappropriate choice could lead to large error in prediction. However, an order betwee 1-3 worked for all problems tested in this thesis.

- The performance of the model under transonic conditions in the presence of moving shocks has been poor compared to subsonic conditions. Under such conditions, the suitability of the method for design optimization is questionable. However, this limitation is not unique to the present approach and other POD based approaches have faced similar issues as pointed out in Chapter 4.

- The number of snapshots required for an acceptable error in prediction was in $\mathcal{O}(100)$. This budget however is well justified in this thesis since the application problems required an order of magnitude more function evaluations.

- The minimum number of samples required in the proposed approach is given by $\varrho = \binom{n+m}{m}$ where n is the interpolation order (typically 1-3) and m is the design dimensionality. Therefore, in high-dimensional systems ($m \to \infty$), the training data size $\to \infty$. In such cases, an *input-space dimensionality reduction* such as *Principal Components Analysis (PCA)* can be used to reduced design parameter dimensionality.

- The methodology is not suitable for extrapolation beyond the original design space from which the snapshots are extracted. Extrapolation violates the fundamental assumption of the model; see Chapter 2.

6.5 Directions of future work

Some key directions in which the future work is expected to progress are listed below

- **Extension to 3D** The present methodology extends to 3D problems without modification in the formulation. The finite volume discretization needs modification to accommodate the 3rd dimension.

- **Extension to Navier-Stokes Equation** This is equivalent to applying the method to different PDE system. While the method generally applies to any PDE system, each new system leads to a unique set of state-observable mapping which completely depends on knowledge of the governing equations in PDE form. As long as this domain knowledge is available, the same method extends to any PDE system.

- **Adaptive sampling of snapshots** An adaptive approach that samples snapshots and updates the ROM development sequentially is required to minimize the snapshot budget. Currently, the snapshot locations are chosen apriori which could lead to a non-optimal distribution - this was particularly true for the transonic test problems. With an adaptive approach, that is goal-oriented such as minimizing error in output predictions, the ROM performance is expected to improve without costing a much higher budget of snapshots.

Appendices

APPENDIX A

MESH MORPHING

In this thesis, we consider geometric shape parameters as design variables. In scenarios where the shape parameters vary, the associated change in geometry under consideration within the flow domain results in an adjustment of the computational mesh. This process is called *mesh morphing*, where the vertices of the computational mesh move while the topology of the mesh (type & number of cells and their arrangement) itself does not change. In order to visualize and/or compute integrated quantities within the flow domain (such as lift/drag/moment about the airfoil), the associated computational grid is necessary; without the spatial location associated, the flow solution data by itself has no meaning. On the other hand, re-constructing the entire computational grid for each new shape parameter is a computationally expensive procedure - consider that along with many-query situations such as design optimization and uncertainty quantification. Therefore in this thesis, a procedure is developed where the computational mesh is morphed every time a new instance of the shape parameters is created. It is done in a way such that it mimics the procedure followed in the high-fidelity CFD package STARCCM+, which is black-box code used to generate snapshots of the FOM in this work [1]. Here, the mesh morphing procedure, associated code development & its validation are provided.

A displacement vector, which could specify components of displacement as dX, dY, dZ is read in by STAR-CCM+ which results in the associated vertices being displaced. Then an interpolation field is generated by training a Radial Basis Function (RBF) [7] for the displacements with respect to the locations of the displacement coordinates. That is for a

[1] See the STAR-CCM+ documentation at `UserGuide>ModelingPhysics>ModelingSpace,Time,andMotion>ModelingMotion>WorkingwithMorphing>MorphingMotionFormulation` for more details

known displacement d_i at each vertex i, it is expanded as

$$d_i = \sum_{j=1}^{N} \phi(r_{ij})\beta_j + \alpha, \;\; i = 1, \ldots, N \tag{A.1}$$

where ϕ is the basis function and r_{ij} is the euclidean distance between two vertices $\mathbf{x}_i$ and $\mathbf{x}_j$, β_j are the coefficients that need to be trained, and α is a constant that serves a purpose similar to an intercept and also needs to be trained. STAR-CCM+ uses a linear basis function, i.e.

$$\phi(r_{ij}) = r_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|_2 \tag{A.2}$$

Additionally, the constraint $\sum_{j=1}^{N} \beta_j = 0$ bounds the magnitude of the coefficients. Therefore, training the RBF is equivalent to solving a linear system of size $(N+1)\times(N+1)$ of the form

$$\begin{bmatrix} \phi(r_{11}) & \ldots & \phi(r_{1N}) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \phi(r_{N1}) & \ldots & \phi(r_{NN}) & 1 \\ 1 & \ldots & 1 & 0 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_N \\ \alpha \end{bmatrix} = \begin{bmatrix} d_1 \\ \vdots \\ d_N \\ 0 \end{bmatrix} \tag{A.3}$$

The solution to Equation A.3 requires computation that scales with N^2. However, unless the entire flow domain is of interest (such as in visualizing a specific solution corresponding to an arbitrary instance of shape parameters), N is typically a small subset of the entire grid. For instance to visualize the pressure distribution about the airfoil, only the vertices that lie on the airfoil are of interest, where N is $<<$ overall grid size.

The procedure is implemented and Figures A.2 and A.1 show that it is validated by comparing against the actual morphed mesh coordinates[2]. The overall purpose of the mesh

[2]For validation purposes, the mesh morphed within STAR-CCM+ for a specific instance of the CST coefficients is exported as a separate CGNS file. The points from this file is extracted to validate the points computed via the trained RBF for the same instance CST coefficients

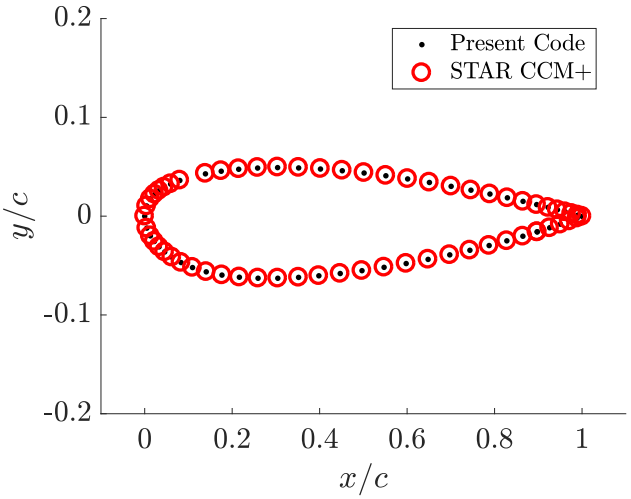


Figure A.1: Validation of the mesh-morphing procedure against STAR-CCM+ for vertices that lie on the airfoil surface

morphing is to compute outputs (such as lift/drag/pressure distributions) and visualization of the solution as shown in Figure A.3.

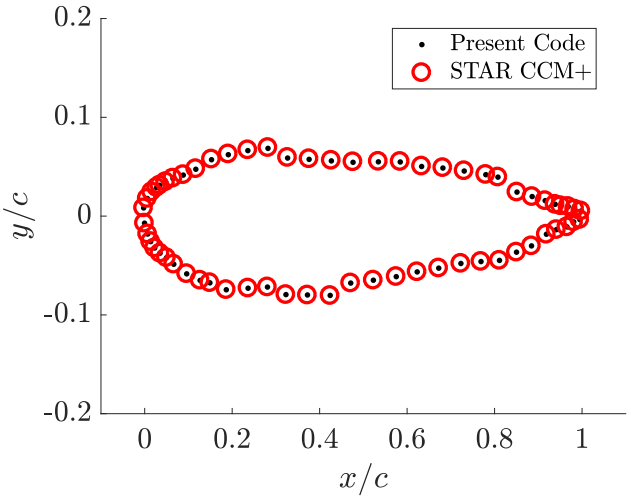


Figure A.2: Validation of the mesh-morphing procedure against STAR-CCM+ for cell-centers that lie immediately adjacent to the airfoil surface

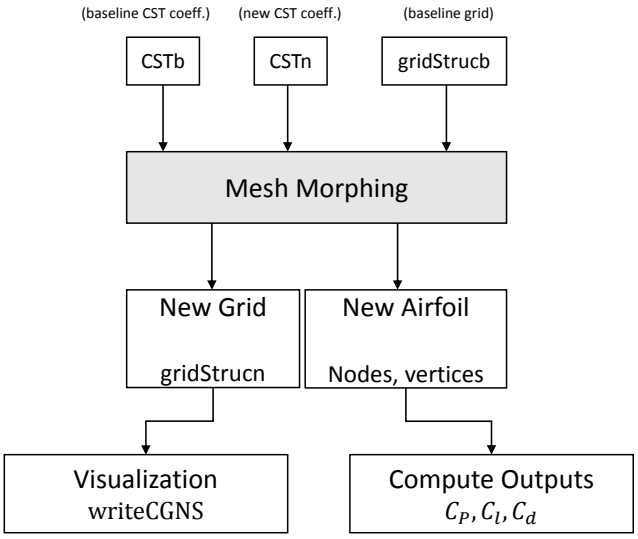


Figure A.3: Flow of information when varying shape parameters that deforms the mesh

APPENDIX B

DISCRETE EMPIRICAL INTERPOLATION METHOD (DEIM)

The Discrete Empirical Interpolation Method (DEIM) is briefly reviewed here and as an illustration one of the non-linear constraints used in the Airfoil test case is evaluated.

For a non-linear function $\mathbf{f}(\theta) \in \mathbb{R}^N$ the DEIM approximates $\mathbf{f}$ by projecting it onto a subspace spanned by $\{\mathbf{x}_1, ..., \mathbf{x}_q\} \subset \mathbb{R}^N$ as

$$\mathbf{f}(\theta) \approx \mathbf{X}c(\theta) \tag{B.1}$$

where $\mathbf{X} = [\mathbf{x_1}, ..., \mathbf{x_q}] \in \mathbb{R}^{N \times q}$, $q << N$ is determined via a POD of the snapshots of $\mathbf{f}$ and is assumed to be globally valid in the design space that bounds the design parameters θ and $\mathbf{c}(\theta) \in \mathbb{R}^q$ are the coefficients of the basis expansion. Then the approximation of $\mathbf{f}$ requires only the determination of $\mathbf{c}(\theta)$ which requires only q equations. The DEIM gives a distinguished set of q points from the over-determined system $\mathbf{f}(\theta) = \mathbf{X}\mathbf{c}(\theta)$. Given a permutation matrix $\mathbf{P}$ that would give q such distinguished rows of a matrix when pre-multiplied, then the $q \times q$ system necessary to solve for the coefficients is given by

$$\mathbf{P}^T\mathbf{f}(\theta) = (\mathbf{P}^T\mathbf{X})\mathbf{c}(\theta) \tag{B.2}$$

So the approximation of $\mathbf{f}(\theta)$ is then given by

$$\mathbf{f}(\theta) \approx \mathbf{X}(\mathbf{P}^T\mathbf{X})^{-1}\mathbf{P}^T\mathbf{f}(\theta) \tag{B.3}$$

If the q row-indices (that are extracted by pre-multiplying with $\mathbf{P}^T$) are represented by a vector, ϱ, then in the above equation, $\mathbf{P}^T\mathbf{f}(\theta)$ is equivalent to extracting the ϱ rows of $\mathbf{f}$. Therefore the approximation of $\mathbf{f}(\theta)$ requires only q computations which is efficient

because $q << N$. Similarly, a non-linear function that depends on the state, $\mathbf{f}(\mathbf{u})$ can be approximated as

$$\mathbf{f}(\mathbf{u}) \approx \mathbf{X}(\mathbf{P}^T\mathbf{X})^{-1}\mathbf{P}^T\mathbf{f}(\mathbf{u}) \tag{B.4}$$

Since $\mathbf{u} = \Phi_k^T\tilde{\mathbf{u}}$ and setting $\tilde{\mathbf{f}} = \Phi_k^T\mathbf{f}(\mathbf{u})$, $\tilde{\mathbf{f}}$ can be approximated as

$$\tilde{\mathbf{f}} = \Phi_k^T\mathbf{X}(\mathbf{P^T}\mathbf{X})^{-1}\mathbf{f}(\mathbf{P}^T\Phi_k\tilde{\mathbf{u}}) \tag{B.5}$$

In the above equation, the term $\Phi_k^T\mathbf{X}(\mathbf{P^T}\mathbf{X})^{-1}$ is independent of the state and hence can be pre-computed and $\mathbf{P}^T\Phi_k$ is just extraction of the ϱ rows of Φ_k. Therefore using the DEIM, the non-linear can be expressed in terms of the reduced state, $\tilde{\mathbf{u}}$ and hence can be efficiently computed.

Now the DEIM is illustrated on evaluating the first constraint of Equation 4.4 which in discretized form is given below

$$\mathbf{h}_1 = \mathbf{y}_5 - \frac{\mathbf{y}_1\mathbf{y}_3}{\mathbf{y}_2} \tag{B.6}$$

Let ϱ_5 be the vector containing the q row-indices returned by DEIM via snapshots of the non-linear term $\mathbf{y}_5$ and ϕ_1, ϕ_2, ϕ_3, ϕ_5 be the projection matrix of $\mathbf{y}_1$, $\mathbf{y}_2$, $\mathbf{y}_3$ and $\mathbf{y}_5$ respectively. Then

$$\tilde{\mathbf{h}}_1 = \tilde{\mathbf{y}_5} - \phi_5^T\mathbf{X}\left[\mathbf{X}(\varrho_5,:)\right]^{-1}\left\{\frac{\phi_1(\varrho_5,:)\tilde{\mathbf{y}_1}\ \ \phi_3(\varrho_5,:)\tilde{\mathbf{y}_3}}{\phi_2(\varrho_5,:)\tilde{\mathbf{y}_2}}\right\} \tag{B.7}$$

In the above equation, the term outside of the braces can be pre-computed. Additionally since $\mathbf{y}_5 = \frac{\mathbf{y}_1\mathbf{y}_3}{\mathbf{y}_2}$, $\mathbf{X} = \phi_5$ and hence the term reduces to $[\mathbf{X}(\varrho_5,:)]^{-1}$ which is $q \times q$ and hence can be cheaply computed. Therefore using the DEIM, the non-linear constraints are evaluated in terms of the reduced state variables which makes it computationally cheap.

APPENDIX C

UNSTRUCTURED GRID HANDLING AND THE CGNS FILE SYSTEM

In this chapter, we briefly review the approach taken to handle reading in the computational grid to compute the linear differential operators.

C.1 The CFD General Notation System

The CGNS [148] is an AIAA-recommended standard for encapsulating CFD data enabling exchange across computing platforms and disciplines. Specifically, the data it encodes include computational grid, boundary conditions and solution in addition to other entities. Due to its wide-spread acceptability among the scientific community across industry and academia and its increasing availability amongst commercial black-box CFD packages, it is the chosen data format upon which this thesis builds its methodology.

The CGNS uses a hierarchical data format called the Standard Interface Data Structures (SIDS) [149] that defines the organization of the data and the associated data structures. The SIDS has a tree-like structure; see Figure C.1 for an illustration of the format where, beginning at a root node several child nodes branch out leading to their own child nodes. Each such node represents a certain entity of the CFD data and/or its properties. For instance, an individual zone of a CFD domain is a node, whose grid information (such as elements connectivity) is a child node which in turn can contain more nodes or data. On the other hand, CGNS also comes with open-source software that provides a framework to create, read and manipulate data in such format. This is available as Application Programming Interface (API) [150] that is available in C++ and Fortran.

In this work we are primarily interested in reading and writing the computational grid in the CGNS format. For this purpose, the unstructured grid capability of the CGNS is used [1].

[1] note that structured grids can be treated a special case of unstructured grids, although CGNS offers

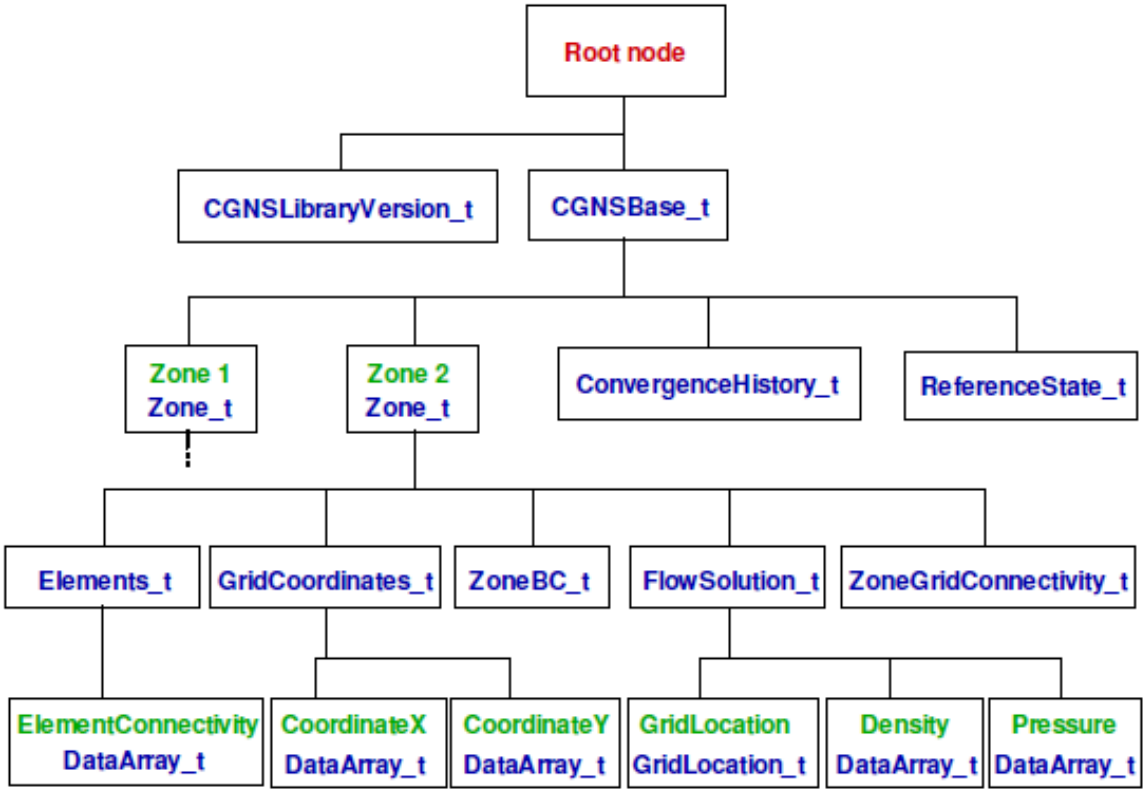


Figure C.1: Tree-like data format of the CGNS [151]

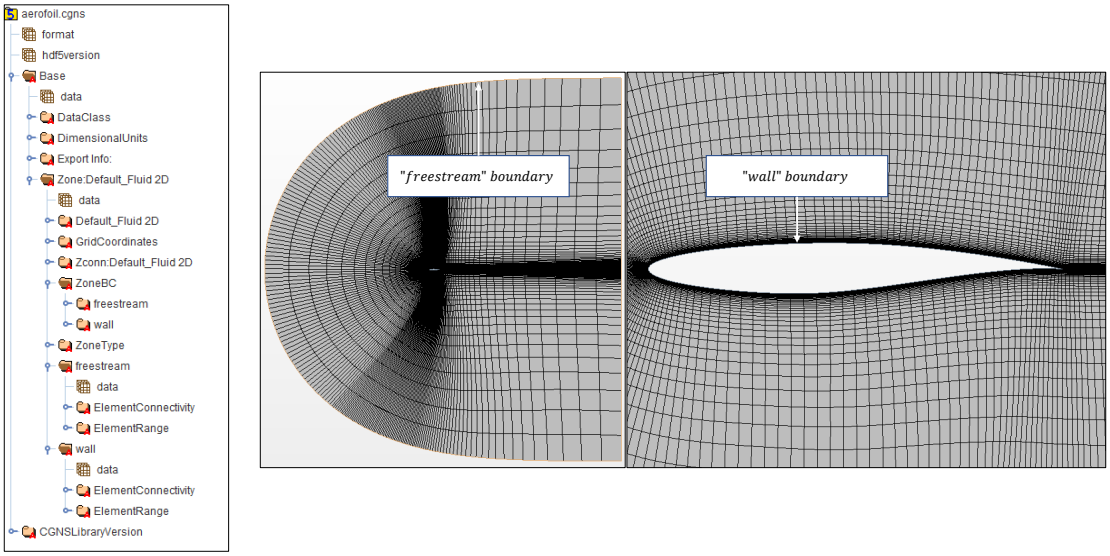


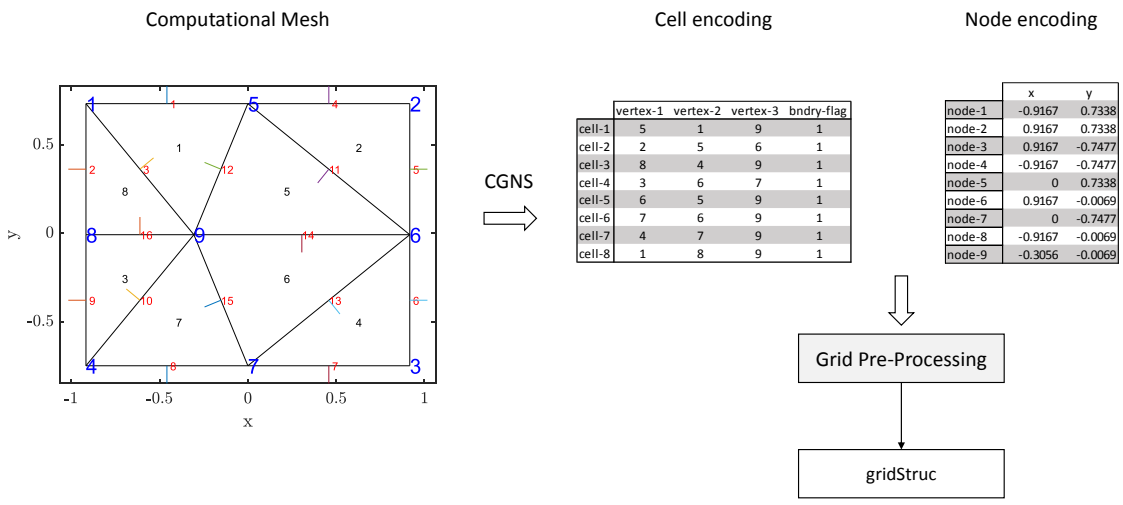
Figure C.2: Example visualization of an airfoil grid in CGNS

See Figure C.2 for an example of the airfoil grid generated in STAR CCM+ and exported in the CGNS format. To the left, a visualization of the of the CGNS file (via a binary file viewer) is shown. Notice the nodes for each of the boundaries and the fluid region, each of which contains the corresponding grid information. This the piece of interest for the present work.

C.2 Unstructured grid encoding

The methodology in this thesis has been developed for unstructured grids which is more common in the high-fidelity analysis of complex geometries. The CGNS format encodes the unstructured grid (in 2 -dimensions) with 2 main pieces of information, namely: (i) the connectivity of each cell with respect to vertices and (ii) the physical (x, y) coordinates of each node in the mesh. Additionally, it stores this information for each entity of the flow domain such as boundaries, fluid regions etc. In this work, the information available in

separate API's for structured grid as well



	vertex-1	vertex-2	vertex-3	bndry-flag
cell-1	5	1	9	1
cell-2	2	5	6	1
cell-3	8	4	9	1
cell-4	3	6	7	1
cell-5	6	5	9	1
cell-6	7	6	9	1
cell-7	4	7	9	1
cell-8	1	8	9	1

	x	y
node-1	-0.9167	0.7338
node-2	0.9167	0.7338
node-3	0.9167	-0.7477
node-4	-0.9167	-0.7477
node-5	0	0.7338
node-6	0.9167	-0.0069
node-7	0	-0.7477
node-8	-0.9167	-0.0069
node-9	-0.3056	-0.0069

Figure C.3: Work flow of the grid pre-processing step based followed in this work before discretizing the linear operator. In the sample grid shown to the left, nodes are represented in blue digits, cell-centers in black and faces in red. The surface normals are shown as short colored line segments starting at the face center

the CGNS format is retrieve via CGNS APIs that gives the cell connectivities and the node coordinates. This information is then *pre-processed* separately within the environment to give the necessary information to discretize the linear terms. The necessary data include connectivites (such as cell-cell, cell-face, node-cell and node-node), surface normals, face area, cell volumes, inter-cell distance, inverse-distance weighted interpolants, cell-center coordinates and face-center coordinates. All of this information is packed into a structure called `gridStruc`. This computation is an expensive process but needs to be done only once - during the online phase of ROM execution the mesh morphing approach explained in Appendix A. The overall process is demonstrated for a simple triangular grid in Figure C.3.

APPENDIX D

COST OF THE FINITE VOLUME DISCRETIZATION STEP

Here we present a more detailed look into the computational cost of approximating linear differential terms using the finite volume method. As mentioned in Chapter 2, although this step is not the most expensive of all the offline computations, it is one of the most important steps in the methodology that overcomes the challenge of using a black-box high-fidelity code. Here we provide details of the computational cost for the laplacian operator in 2D. The cost of other operators such as the gradient is *similar* to that of the laplacian. Further, extending to 3-dimensions costs in the same order of magnitude. Finally, the contents of this chapter are to be read in tandem with Section 2.3.

The construction of the linear differential operator is dependent on knowing the connectivity between the various elements of the mesh. For instance, information about the cells that lie adjacent to a face, the faces that bound a cell, the vertices that make up each edge and the cells that share a vertex are examples of connectivity that encode an unstructured mesh that needs to be known to discretize a PDE on it. Surface normals are also required to compute the component of normal and tangential flux on each cell face. Additionally, the interpolants required to convert between cell-center, face-center and node values are also to be computed. The current section focuses only on estimating the computational complexity and storage for discretizing the linear operator matrix, given all the other aforementioned information since that dominates the overall offline cost of the proposed approach.

We present the computational complexity for the 2-D linear diffusion operator as an illustration. For the exact number of computations, we will have to isolate the interior and boundary cells to compute their individual number of operations and sum them up. Instead, since we are interested only in the order of magnitude of the computational complexity, we focus on computing the complexity for 1 cell and multiply that by the total number of cells,

N. Consider the Equation 2.29 re-written here

$$\sum_f \left(\frac{\Gamma_f A_f}{\delta_f}\right) \left[\underbrace{(u_{k(f)} - u_0)}_{\text{I}} - \underbrace{\left[\frac{u_a - u_b}{|\vec{t}_f|}\right] \hat{t}_f.\vec{l}}_{\text{II}}\right]$$

The above term represents the discretization of the linear diffusion operator that has to be computed for every cell, where for each cell, the summation is over all the faces of the cell. Let N denote the total number of cells, N_f denote the total number of faces, n_f denote the average number of faces per cell, and n_c denote the average number of cells shared by a node.

The term $\left(\frac{\Gamma_f A_f}{\delta_f}\right)$ requires 2 operations and can be pre-computed for all the faces taking a total of $2 \times N_f$ operations. The I term is computed for every face in a cell and its product with the term within parentheses takes a total of $2 \times n_f \times N$. Now focusing on the term II, the dot product $\hat{t}_f.\vec{l}$ takes 3 operations (in 2D) and can be pre-computed for all the faces, leading to a total of $3 \times N_f$. The terms u_a and u_b have to be interpolated from cell-center values and its complexity depends on number of cells shared by a given node. The computation of each of the terms u_a, u_b require $2n_c - 1$ operations. Therefore computing their difference, along with the multiplication and division operations of term II takes a total of $4n_c + 1$ and adding up multiplication with the term within parenthesis makes it $4n_c + 2$ per face and hence a total of $(4n_c + 2) \times n_f \times N$.

In total, the number of operations required to compute the linear diffusion operator is roughly $6N_f + [(4n_c + 4)n_f]N$, where the $6N_f$ arises from pre-computation of terms for every face and $[(4n_c + 4)n_f]N$ arises from computation of terms I and II. The number of cells shared by a node, n_c and the number of faces per cell, n_f are dependent on the mesh. Factoring this as a constant, and considering that the number of faces and cells in a mesh are of the same order of magnitude, the overall complexity is in $\mathcal{O}(N)$. Therefore, the cost of computing matrix $\mathbf{A}$ is linear with respect to the number of cells, N. Note that when $\mathbf{A}$ has parameter dependence, a unique $\mathbf{A}$ has to be computed for each snapshot

and hence for M snapshots, the overall offline cost for computing the linear operators is $\mathcal{O}(MN)$. This is considered as the price paid for the lack of access to the source code of the FOM, which is otherwise essential in the construction of a ROM, and is an offline cost used for model development. Finally, the RHS vector $\mathbf{f}$ needs to be constructed by applying the snapshot vectors to $\mathbf{A}$. Since $\mathbf{A}$ is known to have a highly sparse structure [152], the matrix-vector product would cost $\mathcal{O}(N)$ per snapshot leading to a total of $\mathcal{O}(MN)$. Therefore, the total cost for the offline construction of the matrices for the proposed methodology is still $\mathcal{O}(MN)$.

APPENDIX E

MULTIVARIATE LAGRANGE INTERPOLATION

Below is an example that demonstrate the Algorithm 1 for $m = 2,\ n = 2$. Let $\mathbf{x} = [x_1, x_2]$. Firstly, we pick $\binom{2+2}{2} = 6$ points in the parameter space nearest to the query point $\hat{\mathbf{x}}$. For each point, we want to generate the following polynomials

$$1, x_1,\ x_2,\ x_1x_2, x_1^2,\ x_2^2$$

Each polynomial term in the above equation is represented using the notation $\prod [\mathbf{x}]^{\mathbf{e}_i},\ i = 1, \ldots, 6$. The vectors $\mathbf{e}_i$ are generated via integer partitions. For instance the function (to generate integer partitions) `genPartitions`(m, n) give the following output

`genPartitions`$(2, 2)$ >>

$$\begin{matrix} 0 & 0 \\ 0 & 1 \\ 0 & 2 \\ 1 & 0 \\ 1 & 1 \\ 2 & 0 \end{matrix}$$

where the first row of the output is $\mathbf{e}_1$, second row is $\mathbf{e}_2$ and so on. As an illustration, the term x_2 is approximated as

$$\prod [\mathbf{x}]^{\mathbf{e}_2} = x_1^0 \times x_2^1 = x_2$$

APPENDIX F

ILLUSTRATION OF TANGENT-SPACE INTERPOLATION

F.1 Illustration

The simplest example to consider that demonstrates the application of interpolating on the tangent space to a differential manifold is to consider the following matrix which represents the counter-clockwise rotation of a vector by θ given by

$$\mathbf{A}(\theta) = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix}$$

Such matrices belong to the general orthogonal group $\mathcal{G}(n)$ of orthogonal matrices of size $n \times n$. For $\theta_1 = \frac{\pi}{6}$ and $\theta_2 = \frac{\pi}{3}$ respectively, the matrix is given by

$$\mathbf{A}_1 = \begin{bmatrix} 0.8660 & -0.5000 \\ 0.5000 & 0.8660 \end{bmatrix} \quad \mathbf{A}_2 = \begin{bmatrix} 0.5000 & -0.8660 \\ 0.8660 & 0.5000 \end{bmatrix}$$

now, for an intermediate $\tilde{\theta} = \frac{\pi}{4}$, which linearly interpolates θ_1 and θ_2, a direct linear interpolation of their corresponding matrices gives

$$\mathbf{A}(\tilde{\theta}) = \begin{bmatrix} 0.6830 & -0.6830 \\ 0.6830 & 0.6830 \end{bmatrix} \text{ (incorrect)}$$

The above result is incorrect because we know that

$$\mathbf{A}\left(\frac{\pi}{4}\right) = \begin{bmatrix} 0.7071 & -0.7071 \\ 0.7071 & 0.7071 \end{bmatrix} \text{ (expected)}$$

Now, interpolating the matrix on the tangent space of the manifold, enabled by the expo-

nential and logarithmic mapping defined earlier, leads to the correct result as demonstrated below.

Logarithmic Mapping

$$\Theta_1 = \mathbf{A}(\theta_1) = \begin{bmatrix} 0.0000 & -0.5236 \\ 0.5236 & 0.0000 \end{bmatrix} \; \Theta_2 = \mathbf{A}(\theta_2) = \begin{bmatrix} 0.0000 & -1.0472 \\ 1.0472 & 0.0000 \end{bmatrix}$$

This gives the mapping of the original matrices to the tangential space. Now an interpolation on the tangent space gives

$$\tilde{\Theta} = \begin{bmatrix} 0.0000 & -0.7854 \\ 0.7854 & 0.0000 \end{bmatrix}$$

Exponential Mapping

Now the exponential mapping ($\tilde{\Theta} \rightarrow \tilde{\mathbf{A}}$) gives

$$\tilde{\mathbf{A}} = \begin{bmatrix} 0.7071 & -0.7071 \\ 0.7071 & 0.7071 \end{bmatrix} \text{(correct)}$$

which agrees with the expected result.

REFERENCES

[1] A. I. J. Forrester and A. J. Keane, "Recent advances in surrogate-based optimization," *Progress in Aerospace Sciences*, vol. 45, no. 1-3, pp. 50–79, 2009.

[2] N. V. Queipo, R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P. Kevin Tucker, "Surrogate-based analysis and optimization," *Progress in Aerospace Sciences*, vol. 41, no. 1, pp. 1–28, 2005.

[3] F. A. C. Viana, R. T. Haftka, and L. T. Watson, "Efficient global optimization algorithm assisted by multiple surrogate techniques," *Journal of Global Optimization*, vol. 56, no. 2, pp. 669–689, 2013.

[4] R. H. Myers, D. C. Montgomery, and C. M. Anderson-Cook, *Response surface methodology: process and product optimization using designed experiments*. John Wiley & Sons, 2016.

[5] K. P. Murphy, *Machine learning: a probabilistic perspective*. Cambridge, MA: The MIT Press, 2012.

[6] C. E. Rasmussen, "Gaussian processes in machine learning," in *Advanced lectures on machine learning*, Springer, 2004, pp. 63–71.

[7] M. D. Buhmann, *Radial basis functions: theory and implementations*. Cambridge university press, 2003, vol. 12.

[8] P. B. N. Andy J Keane, *Computational Approaches for Aerospace Design: The Pursuit of Excellence*. WILEY, 2005.

[9] M. A. Grepl, "Reduced-basis approximation a posteriori error estimation for parabolic partial differential equations," PhD thesis, MASSACHUSETTS INSTITUTE OF TECHNOLOGY, 2005.

[10] M. A. Grepl and A. T. Patera, "A POSTERIORI ERROR BOUNDS FOR REDUCED-BASIS APPROXIMATIONS OF PARAMETRIZED PARABOLIC PARTIAL DIFFERENTIAL EQUATIONS," *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 39, no. 1, pp. 157–181, 2005. arXiv: `0608007 [math]`.

[11] K. Veroy and A. T. Patera, "Certified real-time solution of the parametrized steady incompressible Navier-Stokes equations: Rigorous reduced-basis a posteriori error bounds," *International Journal for Numerical Methods in Fluids*, vol. 47, no. 8-9, pp. 773–788, 2005.

[12] M. Drela, "Xfoil: An analysis and design system for low reynolds number airfoils," in *Low Reynolds number aerodynamics*, Springer, 1989, pp. 1–12.

[13] L. R. Miranda, R. D. Elliot, and W. M. Baker, "A generalized vortex lattice method for subsonic and supersonic flow applications," 1977.

[14] N. M. Alexandrov, R. M. Lewis, C. R. Gumbert, L. L. Green, and P. A. Newman, "Approximation and model management in aerodynamic optimization with variable-fidelity models," *Journal of Aircraft*, vol. 38, no. 6, pp. 1093–1101, 2001.

[15] N Alexandrov, R Lewis, C Gumbert, L Green, and P Newman, "Optimization with variable-fidelity models applied to wing design," in *38th Aerospace Sciences Meeting and Exhibit*, 2000, p. 841.

[16] A. Forrester, "Design and analysis of variable fidelity multi-objective experiments," no. September, p. 44, 2011.

[17] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," *Numerische mathematik*, vol. 14, no. 5, pp. 403–420, 1970.

[18] Y. Saad, *Iterative methods for sparse linear systems*. SIAM, 2003.

[19] A. C. Antoulas and D. C. Sorensen, "Approximation of large-scale dynamical systems : An overview," vol. 1892, pp. 1–22, 2001.

[20] P. Benner, S. Gugercin, and K. Willcox, "A Survey of Model Reduction Methods for Parametric Systems," *Max Planck Institute Magdeburg*, 2013.

[21] C. W. Holmes, Philip., Lumley, John L., Berkooz, Gahl and Rowley, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*, 3. 1998, vol. 36, ISBN: 9780521634199.

[22] L. Sirovich, "TURBULENCE AND THE DYNAMICS OF COHERENT STRUCTURES PART III: DYNAMICS AND SCALING," *Quarterly of Applied Mathematics*, vol. XLV, no. 3, pp. 583–590, 1987.

[23] A. C. Antoulas, *Approximation of large-scale dynamical systems*. Siam, 2005, vol. 6.

[24] G. Strang, G. Strang, G. Strang, and G. Strang, *Introduction to linear algebra*. Wellesley-Cambridge Press Wellesley, MA, 1993, vol. 3.

[25] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU Press, 2012, vol. 3.

[26] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *Journal of the society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[27] T Bui-Thanh, "Model-Constrained Optimization Methods for Reduction of Parameterized Large-Scale Systems," PhD thesis, MASSACHUSETTS INSTITUTE OF TECHNOLOGY, 2007.

[28] K. Carlberg, M. Barone, and H. Antil, "Galerkin v. least-squares petrovgalerkin projection in nonlinear model reduction," *Journal of Computational Physics*, vol. 330, no. Supplement C, pp. 693 –734, 2017.

[29] D. Xiao, F. Fang, A. Buchan, C. Pain, I. Navon, J. Du, and G. Hu, "Non-linear model reduction for the NavierStokes equations using residual DEIM method," *Journal of Computational Physics*, vol. 263, pp. 1–18, 2014.

[30] D. Xiao, F.Fang, C. Hu, and G. Hu, "Non-intrusive reduced-order modelling of the NavierStokes equations based on RBF interpolation," *International Journal for Numerical Methods in Fluids*, no. 79, pp. 580–595, 2015. arXiv: `fld.1 [DOI: 10.1002]`.

[31] P. B. N. Christophe Audouze Florian De Vuyst, "Nonintrusive Reduced-Order Modeling of Parametrized Time-Dependent Partial Differential Equations," *Numerical Methods for Partial Differential Equasion*, vol. 29, no. 5, pp. 1587–1628, 2013.

[32] E. N. Lorenz, "Empirical orthogonal functions and statistical weather prediction," 1956.

[33] M. Kirby and L. Sirovich, "Application of the karhunen-loeve procedure for the characterization of human faces," *IEEE Transactions on Pattern analysis and Machine intelligence*, vol. 12, no. 1, pp. 103–108, 1990.

[34] M. D. Graham and I. G. Kevrekidis, "Alternative approaches to the karhunen-loeve decomposition for model reduction and data analysis," *Computers & chemical engineering*, vol. 20, no. 5, pp. 495–506, 1996.

[35] R. Everson and L. Sirovich, "Karhunen–loeve procedure for gappy data," *JOSA A*, vol. 12, no. 8, pp. 1657–1664, 1995.

[36] S. M. Zoldi and H. S. Greenside, "Karhunen-loeve decomposition of extensive chaos," *Physical review letters*, vol. 78, no. 9, p. 1687, 1997.

[37] L Sirovich, "Turbulence and the dynamics of coherent structures. I - Coherent structures. II," *Quarterly of Applied Mathematics (ISSN 0033-569X)*, vol. 45, no. July, p. 561, 1987.

[38] ——, "Turbulence and the dynamics of coherent structures. I - Coherent structures. II - Symmetries and transformations," *Quarterly of Applied Mathematics*, vol. 45, no. 3, pp. 573–582, 1987.

[39] L. Sirovich, "TURBULENCE AND THE DYNAMICS OF COHERENT STRUCTURES PART III: DYNAMICS AND SCALING," *Quarterly of Applied Mathematics*, vol. XLV, no. 3, pp. 583–590, 1987.

[40] C. W. Holmes, Philip., Lumley, John L., Berkooz, Gahl and Rowley, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*, 3. 1998, vol. 36, ISBN: 9780521634199.

[41] A. Chatterjee, "An introduction to the proper orthogonal decomposition," *Current Science*, vol. 78, no. 7, pp. 808–817, 2000.

[42] M. Rathinam and L. R. Petzold, "A New Look at Proper Orthogonal Decomposition," *SIAM Journal on Numerical Analysis*, vol. 41, no. 5, pp. 1893–1925, 2003.

[43] M. Rewienski and J. White, "A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices," *IEEE Transactions on computer-aided design of integrated circuits and systems*, vol. 22, no. 2, pp. 155–170, 2003.

[44] T. Bui-Thanh, M. Damodaran, and K. E. Willcox, "Aerodynamic Data Reconstruction and Inverse Design Using Proper Orthogonal Decomposition," *AIAA Journal*, vol. 42, no. 8, pp. 1505–1516, 2004.

[45] T. Bui-Thanh, M. Damodaran, and K. Willcox, "Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition," *AIAA journal*, vol. 42, no. 8, pp. 1505–1516, 2004.

[46] N. Nguyen, A. Patera, and J Peraire, "A best points interpolation method for efficient approximation of parametrized functions," *International journal for numerical methods in engineering*, vol. 73, no. 4, pp. 521–543, 2008.

[47] S. Chaturantabut and D. C. Sorensen, "Nonlinear Model Reduction via Discrete Empirical Interpolation," *SIAM Journal on Scientific Computing*, vol. 32, no. 5, pp. 2737–2764, 2010.

[48] J. Qian, Y. Wang, H. Song, K. Pant, H. Peabody, J. Ku, and C. D. Butler, "Projection-Based Reduced-Order Modeling for Spacecraft Thermal Analysis," *Journal of Spacecraft and Rockets*, vol. 52, no. 3, pp. 978–989, 2015.

[49] S. A. Renganathan, "A methodology for projection-based model reduction with black-box high-fidelity models," in *17th AIAA Aviation Technology, Integration, and Operations Conference*, 2017, p. 4444.

[50] P. a. Legresley and J. J. Alonso, "Airfoil Design Optimization Using Reduced Order Models Based on Proper Orthogonal Decomposition FLUIDS," *AIAA Fluids 2000 Conference and Exhibit*, 2000.

[51] T. Bui-Thanh, K. Willcox, and O. Ghattas, "Parametric Reduced-Order Models for Probabilistic Analysis of Unsteady Aerodynamic Applications," *AIAA Journal*, vol. 46, no. 10, pp. 2520–2529, 2008.

[52] H. Haddadpour, M. Behbahani-Nejad, and R. D. Firooz-Abadi, "Reduced Order Aerodynamic Model for Aeroelastic Analysis of Complex Configurations in Incompressible Flow," *Journal of Aircraft*, vol. 44, no. 3, pp. 1015–1019, 2007.

[53] T. Lieu and C. Farhat, "Adaptation of Aeroelastic Reduced-Order Models and Application to an F-16 Configuration," *AIAA Journal*, vol. 45, no. 6, pp. 1244–1257, 2007.

[54] K. R. Brouwer, A. R. Crowell, and J. J. Mcnamara, "Rapid Prediction of Unsteady Aeroelastic Loads in Shock-Dominated Flows," no. January, pp. 1–20, 2015.

[55] E. Dowell and K. Hall, "Reduced Order Models in Unsteady Aerodynamic Models, Aeroelasticity and Molecular Dynamics," *ICAS - 26th Congress of International Council of the Aeronautical Sciences 2008*, pp. 1–13, 2006.

[56] B. Glaz, L. Liu, and P. P. Friedmann, "Reduced-Order Nonlinear Unsteady Aerodynamic Modeling Using a Surrogate-Based Recurrence Framework," *AIAA Journal*, vol. 48, no. 10, pp. 2418–2429, 2010.

[57] M. Karpel, "Reduced-Order Models for Integrated Aeroservoelastic Optimization," *Journal of Aircraft*, vol. 36, no. 1, 1999.

[58] G. C. Lewin and H. Haj-Hariri, "Reduced-Order Modeling of a Heaving Airfoil," *AIAA Journal*, vol. 43, no. 2, pp. 270–283, 2005.

[59] T. Skujins and C. E. S. Cesnik, "Reduced-Order Modeling of Unsteady Aerodynamics Across Multiple Mach Regimes," *Journal of Aircraft*, vol. 51, no. 6, pp. 1681–1704, 2014.

[60] J. P. Thomas, E. H. Dowell, and K. C. Hall, "Three-Dimensional Transonic Aeroelasticity Using Proper Orthogonal Decomposition-Based Reduced-Order Models," *Journal of Aircraft*, vol. 40, no. 3, pp. 544–551, 2003.

[61] A. R. Crowell, J. J. Mcnamara, K. M. Kecskemety, and T. W. Goerig, "A Reduced Order Aerothermodynamic Modeling Framework for Hypersonic Aerothermoelasticity," *Direct*, vol. 2969, no. April, p. 22, 2010.

[62] N. J. Falkiewicz, C. E. S. Cesnik, A. R. Crowell, and J. J. McNamara, "Reduced-Order Aerothermoelastic Framework for Hypersonic Vehicle Control Simulation," *AIAA Journal*, vol. 49, no. 8, pp. 1625–1646, 2011.

[63] M. Buffoni and K. Willcox, "Projection-based model reduction for reacting flows," in *40th Fluid Dynamics Conference and Exhibit*, 2010, pp. 1–14, ISBN: 978-1-60086-956-3.

[64] P. C. Chen, D. D. Liu, and K. T. Chang, "TPS Design by a POD / RSM-Based Approach," *Aerospace Engineering*, no. January, pp. 1–24, 2006.

[65] B. I. Epureanu, E. H. Dowell, and K. C. Hall, "Mach Number Influence on Reduced-Order Models of Inviscid Potential Flows in Turbomachinery," *Journal of Fluids Engineering*, vol. 124, no. 4, pp. 977–987, 2002.

[66] B. I. Epureanu, "A parametric analysis of reduced order models of viscous flows in turbomachinery," *Journal of Fluids and Structures*, vol. 17, no. 7, pp. 971–982, 2003.

[67] R. Florea, K. C. Hall, and P. G. a. Cizmas, "Reduced-order modeling of unsteady viscous flow in a compressor cascade," *AIAA Journal*, vol. 36, no. 6, pp. 1039–1048, 1998.

[68] M. Fossati, "Evaluation of Aerodynamic Loads via Reduced-Order Methodology," *AIAA Journal*, vol. 53, no. 8, pp. 1–17, 2015.

[69] M. Gennaretti and L. Greco, "Time-Dependent Coefficient Reduced-Order Model for Unsteady Aerodynamics of Proprotors," *Journal of Aircraft*, vol. 42, no. 1, pp. 138–147, 2005.

[70] M. Gennaretti and D. Muro, "Multiblade Reduced-Order Aerodynamics for State-Space Aeroelastic Modeling of Rotors," *Journal of Aircraft*, vol. 49, no. 2, pp. 495–502, 2012.

[71] D. J. Lucia, P. I. King, and P. S. Beran, "Domain Decomposition for Rediced-Order Modeling of a Flow with Moving Shocks," *AIAA Journal*, vol. 40, no. 11, pp. 2360–2363, 2002.

[72] D. J. Lucia and P. S. Beran, "Projection methods for reduced order models of compressible flows," *Journal of Computational Physics*, vol. 188, no. 1, pp. 252–280, 2003.

[73] D. J. Lucia, P. I. King, and P. S. Beran, "Reduced order modeling of a two-dimensional flow with moving shocks," *Computers and Fluids*, vol. 32, no. 7, pp. 917–938, 2003.

[74] D. J. Lucia, P. S. Beran, and W. a. Silva, "Reduced-order modeling: New approaches for computational physics," *Progress in Aerospace Sciences*, vol. 40, no. 1-2, pp. 51–117, 2004.

[75] G. Biros and O. Ghattas, "Parallel lagrange–newton–krylov–schur methods for pde-constrained optimization. part i: The krylov–schur solver," *SIAM Journal on Scientific Computing*, vol. 27, no. 2, pp. 687–713, 2005.

[76] a. Jameson, "Aerodynamic design via control theory," *Journal of Scientific Computing*, vol. 3, no. 3, pp. 233–260, 1988.

[77] M. B. Giles and N. A. Pierce, "An introduction to the adjoint approach to design," *Flow, Turbulence and Combustion*, vol. 65, no. 3-4, pp. 393–415, 2000.

[78] Z. Lyu and J. R. Martins, "Aerodynamic Design Optimization Studies of a Blended-Wing-Body Aircraft," *Journal of Aircraft*, pp. 1–14, 2014.

[79] S. Chen, Z. Lyu, G. K. W. Kenway, and J. R.R. A. Martins, "Aerodynamic Shape Optimization of the Common Research Model Wing-Body-Tail Configuration," *Journal of Aircraft*, no. January, pp. 1–29, 2015.

[80] S. Shan and G. G. Wang, "Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions," *Structural and Multidisciplinary Optimization*, vol. 41, no. 2, pp. 219–241, 2010.

[81] J. Villemonteix, E. Vazquez, M. Sidorkiewicz, and E. Walter, "Global optimization of expensive-to-evaluate functions: An empirical comparison of two sampling criteria," *Journal of Global Optimization*, vol. 43, pp. 373–389, 2009.

[82] M Björkman and K Holmström, "Global Optimization of Costly Nonconvex Functions Using Radial Basis Functions," *Optimization and Engineering*, vol. 1, no. 4, pp. 373–397, 2000.

[83] A. Cassioli and F. Schoen, "Global optimization of expensive black box problems with a known lower bound," *Journal of Global Optimization*, vol. 57, pp. 177–190, 2013.

[84] H.-M. GUTMANN, "A Radial Basis Function Method for Global Optimization," *Journal of Global Optimization*, vol. 19, pp. 201–227, 2001.

[85] K. Holmström, N. H. Quttineh, and M. M. Edvall, "An adaptive radial basis algorithm (ARBF) for expensive black-box mixed-integer constrained global optimization," *Optimization and Engineering*, vol. 9, no. 4 SPEC. ISS. Pp. 311–339, 2008.

[86] D. Amsallem, M. Zahr, Y. Choi, and C. Farhat, "Design optimization using hyper-reduced-order models," *Structural and Multidisciplinary Optimization*, vol. 51, no. 4, pp. 919–940, 2015.

[87] G. Rozza and A. Manzoni, "Model order reduction by geometrical parametrization for shape optimization in computational fluid dynamics," in *Proceedings of the ECCOMAS CFD 2010, V European Conference on Computational Fluid Dynamics*, 2010.

[88] A. Manzoni, A. Quarteroni, and G. Rozza, "Shape optimization for viscous flows by reduced basis methods and free-form deformation," *International Journal for Numerical Methods in Fluids*, vol. 70, no. 5, pp. 646–670, 2012.

[89] I. Oliveira and A. Patera, "Reduced-basis techniques for rapid reliable optimization of systems described by affinely parametrized coercive elliptic partial differential equations," *Optimization and Engineering*, vol. 8, no. 1, pp. 43–65, 2007.

[90] K. Urban, S. Volkwein, and O. Zeeb, "Greedy sampling using nonlinear optimization," in *Reduced Order Methods for modeling and computational reduction*, Springer, 2014, pp. 137–157.

[91] F. D. V. C. Audouze and P. B. Nair, "Reduced-order modeling of parameterized PDEs using timespace-parameter principal component analysis," *INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING*, vol. 80, pp. 1025–1057, 2009. arXiv: 1010.1724.

[92] C. Prudhomme, D. V. Rovas, K. Veroy, L. Machiels, Y. Maday, A. T. Patera, and G. Turinici, "Reliable Real-Time Solution of Parametrized Partial Differential Equations: Reduced-Basis Output Bound Methods," *Journal of Fluids Engineering*, vol. 124, no. 1, p. 70, 2002.

[93] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, vol. 113, no. 15, pp. 3932–3937, 2016. eprint: http://www.pnas.org/content/113/15/3932.full.pdf.

[94] B. Peherstorfer and K. Willcox, "Dynamic data-driven reduced-order models," *Computer Methods in Applied Mechanics and Engineering*, vol. 291, pp. 21–41, 2015.

[95] B. O. Koopman, "Hamiltonian systems and transformation in hilbert space," *Proceedings of the National Academy of Sciences*, vol. 17, no. 5, pp. 315–318, 1931.

[96] N. Kutz, J. L. Proctor, and S. L. Brunton, "Koopman Theory for Partial Differential Equations," *arXiv:1607.07076v1*, pp. 1–21, 2016. arXiv: arXiv:1607.07076v1.

[97] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, "A Data-Driven Approximation of the Koopman Operator: Extending Dynamic Mode Decomposition," *Journal of Nonlinear Science*, vol. 25, no. 6, pp. 1307–1346, 2015. arXiv: 1312.3019.

[98] S. L. Brunton, B. W. Brunton, J. L. Proctor, and J. N. Kutz, "Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control," *PLoS ONE*, vol. 11, no. 2, pp. 1–19, 2016. arXiv: 1510.03007.

[99] J. N. Kutz, J. L. Proctor, and S. L. Brunton, "GENERALIZING KOOPMAN THEORY TO ALLOW FOR INPUTS AND CONTROL," *arXiv:1602.07647v1*, pp. 1–21, 2016. arXiv: arXiv:1602.07647v1.

[100] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. Kutz, "On dynamic mode decomposition - theory and applications," *Journal of computational dynamics*, no. September, pp. 1–30, 2013. arXiv: arXiv:1312.0041v1.

[101] P. J. SCHMID, "Dynamic mode decomposition of numerical and experimental data," *Journal of Fluid Mechanics*, vol. 656, pp. 5–28, 2010.

[102] A. ALLA, and J. N. Kutz, "NONLINEAR MODEL ORDER REDUCTION VIA DYNAMIC MODE DECOMPOSITION," *SIAM Journal of Scientific Computing*, vol. 39, no. 5, B778–B796, 2017.

[103] Versteeg,H.K., Malalasekera, W., *An Introduction to Computational Fluid Dynamics - The Finite Volume Method*, Pearson, Ed. Prentice Hall, 2007.

[104] W. M. Boothby, *An Introduction to Differentiable Manifolds and Riemannian Geometry*, S. Eilenberg and H. Bass, Eds. Academic Press, 1975, ISBN: 0121160505. arXiv: arXiv:1011.1669v3.

[105] M. Spivak, *Calculus on Manifolds*, R. Gunning and R. Hugo, Eds. Addison-Wesley Publishing Company, 1995.

[106] L. W. Tu, *An Introduction to Manifolds*, S. Axler and K. Ribet, Eds. Springer, 2008, ISBN: 9780387712758.

[107] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, "Riemannian geometry applied to bci classification," in *International Conference on Latent Variable Analysis and Signal Separation*, Springer, 2010, pp. 629–636.

[108] S. Helgason, *Differential geometry and symmetric spaces*. Academic press, 1962, vol. 12.

[109] N. J. Higham, *Functions of matrices: theory and computation*. SIAM, 2008.

[110] N. J. Higham, *Logarithm : Properties , Formulas & Applications*, 2009.

[111] A. H. Al-Mohy and N. J. Higham, "IMPROVED INVERSE SCALING AND SQUARING ALGORITHMS FOR THE MATRIX LOGARITHM," *SIAM J. SCI. COMPUT.*, vol. 34, no. 4, pp. C153–C169, 2012. arXiv: arXiv:1302.5877.

[112] Å. Björck and S. Hammarling, "A schur method for the square root of a matrix," *Linear algebra and its applications*, vol. 52, pp. 127–140, 1983.

[113] N. J. Higham, "Computing real square roots of a real matrix," *Linear Algebra and its applications*, vol. 88, pp. 405–430, 1987.

[114] I. U. Rahman, I. Drori, V. C. Stodden, D. L. Donoho, and P. Schröder, "Multiscale Representations for Manifold-Valued Data," *Multiscale Modeling & Simulation*, vol. 4, no. 4, pp. 1201–1232, 2005.

[115] H. Jeffreys and B. S. Jeffreys, "Methods of mathematical physics," in, 3rd. Cambridge University Press, 1988, ch. Lagrange's Interpolation Formula. P. 260.

[116] M. M. Krady, "Extension of lagrange interpolation," *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH*, 2015.

[117] G. E. Andrews and K. Eriksson, *Integer partitions*. Cambridge University Press, 2004.

[118] K. Schittkowski, "Nlpql: A fortran subroutine solving constrained nonlinear programming problems," *Annals of operations research*, vol. 5, no. 2, pp. 485–500, 1986.

[119] M. Biggs, "Constrained minimization using recursive quadratic programming," *Towards global optimization*, 1975.

[120] S.-P. Han, "A globally convergent method for nonlinear programming," *Journal of optimization theory and applications*, vol. 22, no. 3, pp. 297–309, 1977.

[121] M. J. Powell, "A fast algorithm for nonlinearly constrained optimization calculations," in *Numerical analysis*, Springer, 1978, pp. 144–157.

[122] R. Fletcher, "Practical methods of optimization john wiley & sons," *New York*, vol. 80, 1987.

[123] P. E. Gill, W. Murray, and M. H. Wright, "Practical optimization," 1981.

[124] J. Nocedal and S. J. Wright, *Sequential quadratic programming*. Springer, 2006.

[125] *Matlab pde toolbox release r2017b, the mathworks inc.* 2017.

[126] A. T. Patera, "Reduced Basis Approximation and A Posteriori Error Estimation for Parametrized Partial Differential Equations," *Foundations*, no. January, p. 251, 2007.

[127] J. V. Joris Degroote and K. Willcox, "Interpolation among reduced-order matrices to obtain parameterized models for design, optimization and probabilistic analysis," *International Journal for Numerical Methods in Fluids*, vol. 63, pp. 207–230, 2010. arXiv: `fld.1 [DOI: 10.1002]`.

[128] D Amsallem and C Farhat, "Interpolation Method for Adapting Reduced-Order Models and Application to Aeroelasticity," *AIAA Journal*, vol. 46, no. 7, pp. 1803–1813, 2008.

[129] (2017). Starccm+ url: Http://mdx.plm.automation.siemens.com/star-ccm-plus.

[130] D. Poirier, S. R. Allmaras, D. R. McCarthy, M. F. Smith, and F. Y. Enomoto, "The cgns system," *AIAA paper*, no. 98-3007, 1998.

[131] D Kraft and K Schnepper, "Slsqp a nonlinear programming method with quadratic programming subproblems," *DLR, Oberpfaffenhofen*, 1989.

[132] M. S. Selig, *UIUC airfoil data site*. Department of Aeronautical and Astronautical Engineering University of Illinois at Urbana-Champaign, 1996.

[133] D. J. Lucia, P. S. Beran, and W. a. Silva, "Reduced-order modeling: New approaches for computational physics," *Progress in Aerospace Sciences*, vol. 40, no. 1-2, pp. 51–117, 2004.

[134] P. A. LeGresley, *Application of proper orthogonal decomposition (POD) to design decomposition methods*. Stanford University.

[135] T. J. Santner, B. J. Williams, and W. I. Notz, *The design and analysis of computer experiments*. Springer Science & Business Media, 2013.

[136] B. Kulfan and J. Bussoletti, “” fundamental” parameteric geometry representations for aircraft component shapes,” in *11th AIAA/ISSMO multidisciplinary analysis and optimization conference*, 2006, p. 6948.

[137] B. M. Kulfan, “Universal parametric geometry representation method,” *Journal of Aircraft*, vol. 45, no. 1, pp. 142–158, 2008.

[138] J. R.R. A. Martins, “Wing Design via Numerical Optimization Optimization Problem Computational Models,” *SIAG/OPT Views and News*, vol. 23, no. 1, pp. 2–7, 2015.

[139] M. Drela, “Pros and Cons of Airfoil Optimization,” in *Proceedings of Frontiers of Computational Fluid Dynamics*, 1998, ISBN: 9810237073.

[140] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989, ISBN: 0201157675.

[141] J. Kennedy, “Particle swarm optimization,” in *Encyclopedia of machine learning*, Springer, 2011, pp. 760–766.

[142] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, “Optimization by simulated annealing,” *science*, vol. 220, no. 4598, pp. 671–680, 1983.

[143] O. Chernukhin and D. W. Zingg, “Multimodality and Global Optimization in Aerodynamic Design,” *AIAA Journal*, vol. 51, no. 6, pp. 1342–1354, 2013.

[144] J. D. Anderson Jr, *Fundamentals of aerodynamics*. Tata McGraw-Hill Education, 2010.

[145] C. N. Adkins and R. H. Liebeck, “Design of optimum propellers,” *Journal of Propulsion and Power*, vol. 10, no. 5, pp. 676–682, 1994.

[146] S. A. Renganathan, R. K. Denney, A. Duquerrois, and D. N. Mavris, “Validation and assesment of lower order aerodynamics based design of ram air turbines,” in *12th International Energy Conversion Engineering Conference*, 2014, p. 3463.

[147] C. Z. Mooney, *Monte carlo simulation*. Sage Publications, 1997, vol. 116.

[148] D. Poirier, S. Allmaras, D. McCarthy, M. Smith, and F. Enomoto, “The cgns system,” in *29th AIAA, Fluid Dynamics Conference*, 1998, p. 3007.

[149] S Allmaras, “Cgns standard interface data structures,” *Draft, May*, 1997.

[150] (). Cfd general notation system standard interface data structures.

[151] C. L. Rumsey, D. M. Poirier, R. H. Bush, and C. E. Towne, "A user's guide to cgns," 2001.

[152] S. A. Renganathan and D. N. Mavris, "A methodology for projection-based model reduction with black-box high-fidelity models," in *17th AIAA Aviation Technology, Integration, and Operations Conference*, 2017, p. 4444.